POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Detection of Illegal Landfills using Deep Learning: A Weakly Supervised Approach

Tesi di Laurea Magistrale in
Computer Science and Engineering - Ingegneria Informatica

Author: **Simone Sorrenti**

Student ID: 968282
Advisor: Prof. Giacomo Boracchi
Co-advisors: Prof. Piero Fraternali, Ph.D. Andrea Diecidue
Academic Year: 2023-24

Abstract

In our current era, the urgency to protect our environment has intensified, necessitating swift action. The rise of illegal landfill sites presents imminent ecological risks and long-term consequences for ecosystems and human health. Detecting and dismantling these covert waste sites is crucial in combating organized environmental crimes. Our goal is to utilize deep learning techniques to automatically locate and outline illegal landfills in satellite images. This thesis contributes to the European initiative PERIVALLON, aimed at preventing unlawful activities like unauthorized waste disposal.

To tackle the problem of identifying and localizing illegal dumpsites, given the scarcity of waste-containing images we employed a weakly supervised segmentation approach. Specifically, we compared two popular approaches: heatmap-based and MIL-based. In the heatmap-based approach, we employed a CNN trained to classify waste presence or absence in an image. Post-training, we derived a heatmap using GRAD-CAM++ from the CNN's final feature maps. Alternatively, the model directly generated a heatmap using features from different layers at varying resolutions. In the MIL-based approach, we trained a CNN to classify small patches extracted from the high-resolution input image. Each patch was processed to obtain representations, aggregated using pooling to determine importance, and produce a single image representation fed into a classification head. The trained model can classify bags of patches of variable size, generating a heatmap by extracting and classifying patches with a reduced shift. Moreover, given the diverse sources of satellite imagery, we standardized them to uniform spatial resolutions and dimensions through preprocessing and created a data augmentation framework to simulate satellite conditions, including increased brightness and added noise to images.

Keywords: illegal landfill detection, deep learning techniques, satellite image analysis, weakly supervised segmentation, convolutional neural networks, multiple instance learning

Abstract in lingua italiana

Nell'attuale contesto, la crescente urgenza di preservare l'ambiente richiede azioni tempestive. L'aumento delle discariche illegali rappresenta una minaccia ecologica imminente con implicazioni a lungo termine per gli ecosistemi e la salute umana. Il nostro obiettivo è impiegare tecniche di deep learning per rilevare e delineare automaticamente le discariche illegali nelle immagini satellitari, contribuendo all'iniziativa europea PERIVALLON contro il crimine ambientale.

Per affrontare il problema della localizzazione dei siti illegali e data la scarsità di immagini contenenti rifiuti, abbiamo utilizzato un approccio di segmentazione debolmente supervisionato. In particolare, abbiamo confrontato due approcci popolari: basati sulle mappe di calore e basati sul Multiple Instance Learning (MIL). Nel metodo basato sulle mappe di calore, abbiamo utilizzato una rete neurale convoluzionale (CNN) addestrata per classificare un'immagine. Dopo l'addestramento, abbiamo derivato una mappa di calore utilizzando GRAD-CAM++ dalle ultime mappe di attivazione della CNN. In alternativa, il modello ha generato direttamente una mappa di calore combinando le mappe provenienti da diversi livelli. Nel metodo basato sul MIL, abbiamo addestrato una CNN per classificare piccole patch estratte dall'immagine ad alta risoluzione. Ogni patch è stata elaborata per ottenere delle rappresentazioni, aggregate tramite pooling per determinarne l'importanza e produrre una singola rappresentazione dell'immagine per poterla classificare. Il modello addestrato può classificare insiemi di patch di dimensioni variabili, generando una mappa di calore estraendo e classificando patch con uno spostamento ridotto. Inoltre, data la diversità delle fonti di immagini satellitari, abbiamo standardizzato tali immagini a risoluzioni spaziali e dimensioni uniformi mediante pre-elaborazione ed abbiamo sviluppato un framework di data augmentation per simulare le condizioni satellitari, incluso l'aumento della luminosità e l'aggiunta di rumore alle immagini.

Parole chiave: rilevamento di discariche illegali, tecniche di apprendimento profondo, analisi di immagini satellitari, segmentazione supervisionata debolmente, reti neurali convoluzionali, apprendimento a istanze multiple

Contents

Introduction

In recent years, the global increase in industrial production has led to a significant rise in waste generation, with far-reaching implications for human health, environmental preservation, sustainability, and the circular economy. It's important to note that outside the established legal frameworks, developing countries often face even greater challenges in waste management, with disastrous consequences for local communities and surrounding ecosystems [13]. The environmental consequences of inadequately managed waste are diverse and extremely damaging. Soil and water contamination can compromise the quality of natural resources and jeopardize food safety, with severe implications for human health [1]. Wildlife is at risk of ingesting or becoming trapped in waste, enduring suffering and death, while plants can be harmed by soil contamination. Moreover, greenhouse gas emissions generated by the decomposition of organic waste contribute to global warming and climate change [37]. Not to be overlooked is also the pollution of groundwater, caused by the washing of pollutants into the subsoil through rainfall, as well as the dispersion of gaseous pollutants into the air through wind. These phenomena can have a significant impact on air, land, and water quality, with serious repercussions for human health and the environment. Over recent decades, there has been an undeniable surge in global industrial output, leading to a concerning rise in waste production. This issue of waste management extends far beyond mere environmental concerns, touching upon crucial aspects of public health, sustainable development, and economic viability. The improper disposal of waste, often facilitated by illegal dumping practices, poses a multifaceted threat to our world. Not only does it pollute our air, soil, and water sources, but it also disrupts fragile ecosystems and endangers biodiversity.

While the issue of illegal waste disposal has long plagued communities globally, its detection remains a significant challenge in the modern era. Clandestine landfills, often concealed from plain sight, pose substantial environmental risks and endanger public health. However, to confront this issue head-on, innovative strategies are imperative. Building upon the urgency to address illegal waste dumping, recent research endeavors, such as the European PERIVALLON project[11, 42], have emerged. This project not only aims to detect and localize illegal landfills but also aligns with a pivotal initiative striv-

ing to combat organized environmental crime through the application of state-of-the-art artificial intelligence solutions. For instance, leveraging satellite imagery and advanced binary classification and segmentation techniques, researchers are able to identify and map out these illicit dumping sites, thus contributing to global efforts for a sustainable and environmentally conscious future.

While traditional on-site inspections remain essential for evaluating the extent of illegal landfill activities, monitoring vast territories poses significant challenges. Recent studies have demonstrated that advanced technologies for aerial imagery analysis provide a promising avenue, enabling swift identification of potential illegal landfill sites. In addition, the utilization of Computer Vision methodologies, particularly Deep Learning, aids in the advancement of automated solutions proficient in identifying nuanced features of landfills, thereby amplifying the effectiveness of the detection process, as highlighted in a recent survey on solid waste detection in remote sensing images [14].

However, the scarcity of finely annotated data presents a significant hurdle in training sophisticated deep learning models. This limitation is particularly pronounced in positive images depicting the presence of waste, as annotating these images involves a laborious and intricate process by humans. Moreover, annotating for precise localization of waste, such as bounding boxes or segmentation masks, proves even more challenging, especially in satellite imagery. Consequently, datasets often lack detailed annotations at the image-level for positive images and lack any annotations for waste localization or segmentation. To circumvent this limitation, innovative approaches like Weak and Self Supervision and Data Augmentation techniques have emerged, offering viable alternatives for training models with limited annotated data.

Another well-documented challenge associated with satellite imagery is the presence of small object instances dispersed throughout images. Furthermore, backgrounds depicted in these images often exhibit high complexity, with multiple ground objects coexisting. Additionally, satellite imagery showcases a diverse range of object sizes, varying from compact entities to expansive features. Moreover, objects within these images can adopt arbitrary orientations, adding complexity to the tasks of detection and classification. Finally, satellite imagery commonly displays high intra-class diversity and notable inter-class similarity, rendering the differentiation between various object classes particularly challenging [30].

My contribution encompasses three primary research points. Firstly, we dedicated attention to optimizing image pre-processing techniques. This involved working with the AerialWaste dataset [53], which contains images from diverse sources with varying pixel

dimensions and spatial resolutions (expressed in cm/pixel). Standardizing the images to uniform spatial resolutions and dimensions was crucial to aid the models in identifying objects consistently at the same scale. Furthermore, considering that certain Convolutional Neural Networks (CNNs) demonstrate limitations with specific resolutions, selecting the most suitable spatial resolution was paramount based on these insights.

Secondly, we addressed the data augmentation process. Given the limited availability of positive labels within the dataset, employing data augmentation was essential to expand the dataset and enhance the model's performance. We endeavored to implement a data augmentation framework that generated images closely resembling satellite reality, incorporating factors such as simulated increased brightness or the addition of noise to the image.

Lastly, we aimed to compare two of the most popular techniques used for weakly supervised localization/segmentation in the literature: CAM-based and MIL-based [59]. CAM (Class Activation Map) and MIL (Multiple Instance Learning) are widely used approaches for identifying regions of interest in images through weakly supervised learning.

CAM relies on the concept of using class activation maps produced by a convolutional neural network to identify regions of interest in the image. Regions with higher activations are considered as regions of interest for that class. On the other hand, MIL is based on learning from sets of instances rather than individual examples. During training, the model seeks to learn which sets of instances are associated with a particular class, without knowing the specific label for each instance.

The main difference between CAM-based and MIL-based lies in how they handle label information during training. While CAM uses label information to generate class activation maps, MIL assumes that labels are only available for sets of instances and seeks to learn associations between these sets and classes without knowing the specific label for each instance.

In the heatmap-based approach, we employed a CNN trained to distinguish between the presence and absence of waste in an image. Following training, we generated a heatmap using GRAD-CAM++ from the final feature maps of the CNN. However, due to the lower resolution of these final features compared to the original image, we achieved good localization but poor definition of waste boundaries. As an alternative, the model can directly generate a heatmap by solely applying convolution operations, bypassing Dense layers for classification, as probabilities were determined through special pooling directly from the heatmaps. These heatmaps were generated using features from different layers at various resolutions. This approach allowed us to utilize early layers for more accurate

contour definition and later layers for enhanced localization, thereby improving waste localization results. The final heatmap was formed by combining heatmaps generated at different resolutions. Additionally, to encourage sparsity of positive pixels, indicating the presence of waste, an L1 regularization term was applied to the generated heatmaps, as positive pixels are much less prevalent compared to negative background pixels.

Meanwhile, in the MIL-based approach, we trained a CNN to classify a collection of very small relevant patches extracted from the high-resolution input image. These patches can be extracted using various methodologies, with a common approach being to apply a grid where each cell has a size of PxP, and each grid represents a non-overlapping extractable patch. Given these non-overlapping patches possible from the grid, it's possible to use all of them or only a subset. When deciding to use a subset, it's essential to determine a good approach to extract relevant patches that will constitute the bag upon which training or inference will occur. Each patch is processed by the CNN to obtain patch representations. These representations are then aggregated using pooling to determine their importance and produce a single image representation, which is provided to a classification head. After training the model in this manner, it can classify bags of patches of any size, including single patches, thus generating a heatmap by extracting patches with reduced shift. Pixels within each patch are then assigned their probability or importance weight, or a fusion of the two. In overlapping patches, pixel values are summed and averaged across patches to generate an average score.

We organise the thesis work in the following chapters:

- Chapter 1: In this chapter, we formally state the formulation of our problem, delineate our approach, and outline the characteristics of our work.
- Chapter 2: This chapter covers the theoretical background of our work. We delve into the theory behind deep learning and its various tasks, including Convolutional Neural Networks (CNNs), which are commonly used for image analysis. Additionally, we discuss the most popular approaches used in supervised and weakly supervised localization and segmentation. Specifically, we focus on cam-based and mil-based approaches for the latter. Finally, we analyze the challenges posed by a dataset of satellite images and review how some previous works have addressed similar tasks.
- Chapter 3: This chapter presents the dataset used in our study and the solutions proposed to address the task of waste localization through weakly supervised segmentation.

- Chapter 4: This chapter presents the results of the experiments conducted on our model.
- Chapter 5: The concluding chapter summarizes our work and provides insights into potential avenues for future research.

1 | Problem Formulation

Detecting the presence of waste in a piece of land, and thus within an area, through binary classification is useful, but semantic segmentation proves even more valuable. Semantic segmentation enables us to identify different types of waste within the area, particularly if multiple wastes are present. Moreover, it allows us to delineate the boundaries of these waste types, facilitating more precise and accurate localization.

The problem of semantic segmentation in waste detection can be formally described as follows:

Given an image $I \in \mathbb{R}^{W \times H \times C}$ and a set of classes $C = \{c_1, c_2, .., c_k\}$, find a function $g : \mathbb{R}^{W \times H \times C} \rightarrow \mathbb{R}^{W \times H}$ that associates to each pixel a unique class $c_i \in C$. The function g generates a mask $\Delta \in \mathbb{R}^{W \times H}$ in which for each pixel we have a unique value that represents the class to which it belongs.

The dataset used in this study is the AerialWaste dataset [53], which comprises satellite images of the Lombardia region provided by the ARPA agency in Italy. In its version 2.0, the dataset includes a total of 10,977 RGB satellite images, meticulously divided into training and testing sets to facilitate precise comparisons between various studies. Among these images, there are 7,318 negative samples and 3,659 positive samples. The images available do not all come from the same source; rather, we can identify three different sources: AGEA Ortophotos, WorldView-3, and GoogleEarth. Depending on the source, the images have different sizes and spatial resolutions (cm/pixel).

This problem was approached considering various relevant aspects to determine the most suitable approach to pursue for the study, including the scarcity of annotations, both positive images containing waste and segmentation masks, leading us to adopt weakly supervised training to localize the waste. Additionally, images from different sources are available with varying pixel dimensions and spatial resolution (cm/pixel), thus necessitating standardization to a uniform dimension and spatial resolution to facilitate model learning and subsequent inference.

Figure 1.1: Some images extracted from the AerialWaste dataset containing waste in the area.

Our involvement lies within the PERIVALLON initiative [11, 42], which is supported by funding from the European Union. This project is dedicated to addressing the pressing issue of organized environmental crime, which encompasses activities such as deliberate pollution and illegal disposal of various types of waste, presenting intricate challenges for detection and investigation. The primary goal of the project is to combat organized environmental crime by advancing innovative tools and solutions, as well as fostering international cooperation. Through the utilization of state-of-the-art technologies including artificial intelligence, computer vision, geospatial intelligence, remote sensing, and online monitoring, PERIVALLON aims to enhance investigation procedures and establish an Environmental Crime Observatory. This observatory is designed to proactively prevent environmental crimes throughout Europe. With a diverse consortium consisting of law enforcement agencies, academic institutions, and industry partners, PERIVALLON is well-positioned to effectively address the complex challenges posed by organized environmental crime.

Furthermore, to conduct this study, we paid attention to and referenced a previous study conducted within the PERIVALLON project, titled "AerialWaste dataset for landfill discovery in aerial and satellite images" [53]. In this study, the problem of binary classification, determining the presence or absence of waste in a satellite image on the AerialWaste dataset version 1.0, was addressed. The binary classifier achieved an 80.70% F1 score,

with 81.89% precision at 79.54% recall on the test set. It is important for us to consider this work, especially the backbone utilized (ResNet50), to continue our study and identify possible limitations of this backbone. Moreover, aligning with this work allows us to understand if improvements can be achieved with different architectures centered around the ResNet50 backbone.

In particular, in the previous investigation aimed at addressing the challenge of waste classification and localization, a model was trained employing a specific architecture outlined in Figure 1.2. This architecture featured the utilization of a Convolutional Neural Network (CNN), specifically the ResNet50, serving as the backbone. Subsequently, the Feature Pyramid Network (FPN) architecture was applied to enhance object detection performance. FPN facilitates improved detection, especially when dealing with objects of varying scales, by integrating low-resolution semantically robust features with high-resolution semantically weaker ones. This integration is achieved through a top-down pathway that complements the conventional bottom-up feature extraction process of CNNs, enabling the extraction, adaptation, and fusion of features at multiple levels. During the training phase, data augmentations such as flipping, rotation, and cropping were implemented. Input images were resized to a fixed dimension to accommodate various sizes within the same batch. After the final fully connected (FC) classification layer, a Sigmoid function was applied to generate a confidence score ranging from 0 to 1, indicating the model's certainty regarding the image belonging to the positive class. Additional details regarding the training methodology are accessible in the model's Git repository. A classification threshold of 0.5 was employed to categorize each image based on this confidence score.

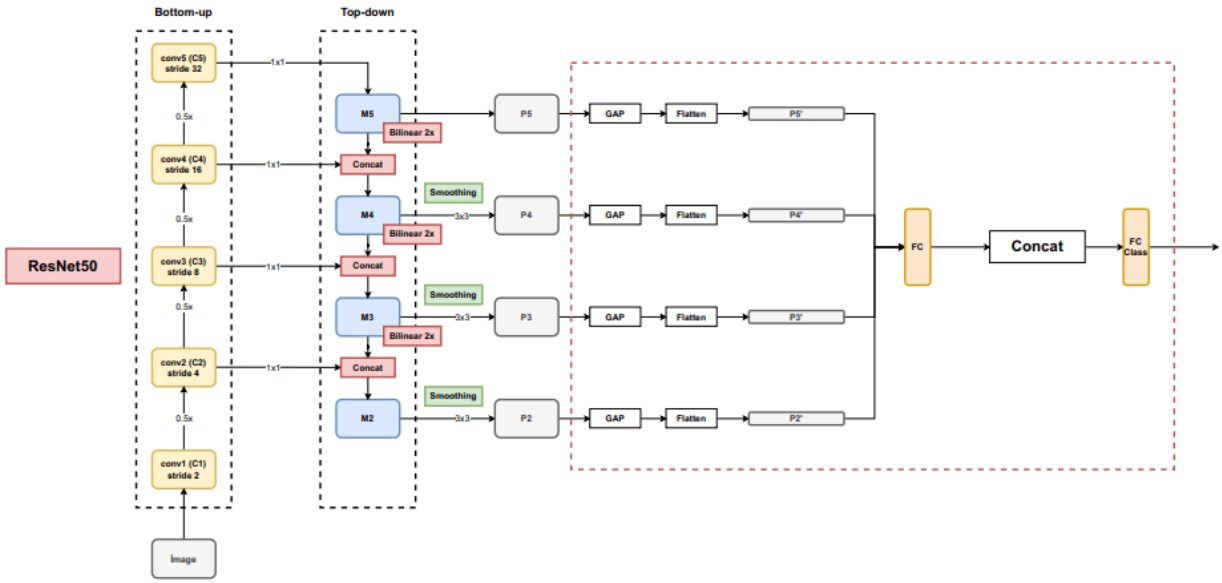


Figure 1.2: Some images extracted from the AerialWaste dataset containing waste in the area.

2 | Background

2.1. Deep Learning

The advent of deep learning has catalyzed profound transformations across various domains of artificial intelligence (AI), empowering systems to tackle complex tasks with unprecedented accuracy and efficiency. Rooted in the paradigm of artificial neural networks, deep learning architectures, characterized by their multi-layered structure, have emerged as formidable tools for learning intricate patterns and representations from vast datasets.

In this section, we embark on a journey to explore the landscape of deep learning, focusing on two key aspects: the spectrum of tasks that deep learning can address for Computer Vision domain and the evolution of two prominent architectures, Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs).

Before delving into specific architectures, it's crucial to understand the diverse tasks that deep learning can proficiently handle. From image classification and object detection to natural language processing and speech recognition, deep learning algorithms have demonstrated remarkable versatility across a wide array of domains. We will examine these tasks in detail, elucidating their objectives, challenges, and real-world applications.

One of the foundational pillars of deep learning, CNNs have revolutionized the field of computer vision. By leveraging specialized layers such as convolutional and pooling layers, CNNs excel at extracting hierarchical features from images, enabling tasks such as image classification, object detection, and semantic segmentation. We will delve into the architecture and workings of CNNs, exploring their evolution, key components, and state-of-the-art advancements.

In recent years, Vision Transformers have emerged as a disruptive force in the field of computer vision, challenging the supremacy of CNNs. Unlike traditional CNNs, ViTs adopt a transformer architecture, originally designed for sequential data processing in natural language processing tasks. We will investigate the principles behind ViTs, their

architecture, and their efficiency in tasks such as image classification and object detection, as well as their potential advantages over CNNs.

Through this exploration, we aim to provide a comprehensive understanding of both the breadth of tasks that deep learning can tackle and the evolution of two prominent architectures, CNNs and ViTs. By the end of this section, readers will gain insights into the capabilities and nuances of these architectures, empowering them to make informed decisions in applying deep learning to their specific domains.

2.1.1. Task

Image processing is a captivating and ever-evolving field of artificial intelligence, focusing on the analysis, manipulation, and interpretation of digital images. Among the various challenges addressed in this domain, several key tasks stand out as the foundation of deep learning applications in computer vision. It is essential to understand the different tasks that can be tackled to handle images accurately and effectively. Among the most relevant are image classification, object detection, object segmentation, image style transfer, image colorization, image reconstruction, image super-resolution, and image synthesis. Each task has its own characteristics and specific applications, and a thorough study of them is crucial for developing innovative solutions in multiple sectors, from industrial automation to medical diagnostics, from video surveillance to artistic creation. We will delve into some of the most important tasks mentioned earlier, which are relevant to my work, outlining their main features and providing concrete examples of their real-world applications.

Image Classification The task of image classification, also known as object recognition or image labeling, involves assigning a label or tag to an entire image based on existing training data of labeled images. Although it may seem like a straightforward process, it requires detailed analysis of individual pixels to determine the most suitable label for the entire image, providing valuable insights and enabling informed decisions.

There are various methodologies for image classification tailored to different problem scenarios:

- Binary Classification: This method categorizes images into two distinct classes, making decisions based on an "either/or" logic. For example, it can be used to distinguish between benign and malignant tumors in medical images or to detect defects in product quality.
- Multiclass Classification: Unlike binary classification, multiclass classification categorizes images into three or more classes. It finds applications in various fields,

such as sentiment analysis in natural language processing or disease classification in medical diagnosis.

- Multilabel Classification: Unlike multiclass classification, where each image is assigned to a single class, multilabel classification allows for the assignment of multiple labels to an image. For example, an image depicting a fruit salad may be labeled with various colors.

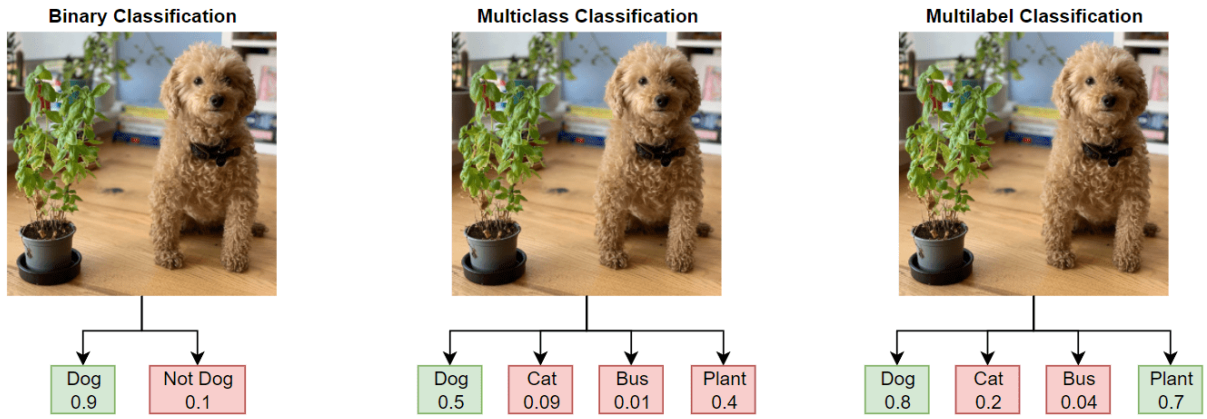


Figure 2.1: Illustration of different approaches to image classification: binary, multi-class, and multi-label.

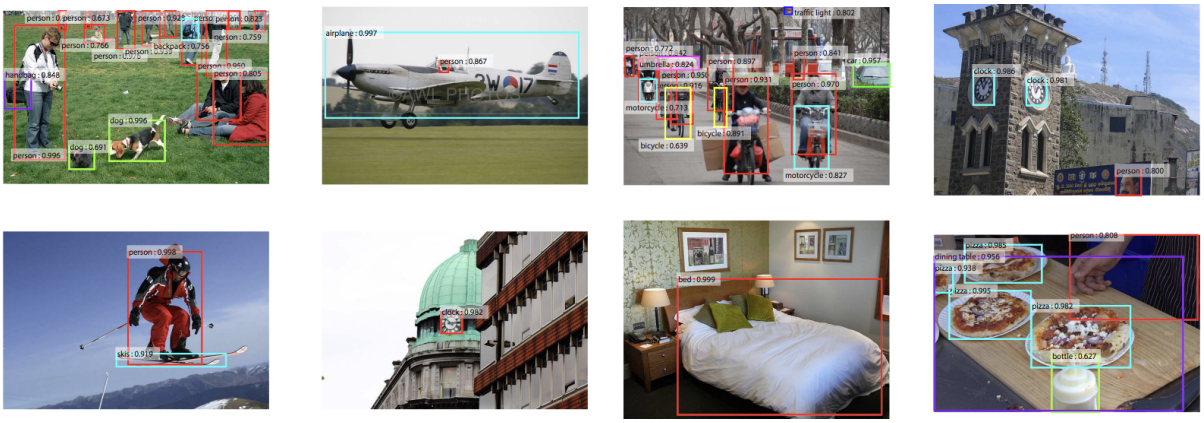


Figure 2.2: Example of Object Detection With Faster R-CNN [46] on the MS COCO Dataset.[31]

Object Detection Object detection encompasses both image classification and localization, as it involves identifying and labeling multiple objects within an image. This task is more complex than simple image classification, as it requires accurately localizing and classifying objects of varying types within the image. Examples of object detection tasks

include delineating bounding boxes and assigning labels to objects within various scenes, such as street scenes, indoor environments, and landscapes.

Object segmentation Object segmentation, also referred to as semantic segmentation, holds significant importance in computer vision tasks by accurately outlining object boundaries within an image. Unlike object detection, which primarily entails outlining bounding boxes around objects, object segmentation goes beyond by identifying the specific pixels associated with each object. This meticulous localization offers a more nuanced comprehension of the objects depicted in the image.

Two primary types of image segmentation are recognized:

- Semantic Segmentation: This method entails assigning a label to every pixel in the image, facilitating detailed object classification. In contrast to classification tasks where a single label is applied to the entire image, semantic segmentation treats multiple objects of the same class as a unified entity.
- Instance Segmentation: Building upon semantic segmentation, instance segmentation further distinguishes between individual instances of objects within the same class. Typically more intricate than semantic segmentation, instance segmentation necessitates accurately delineating each object instance within the image.

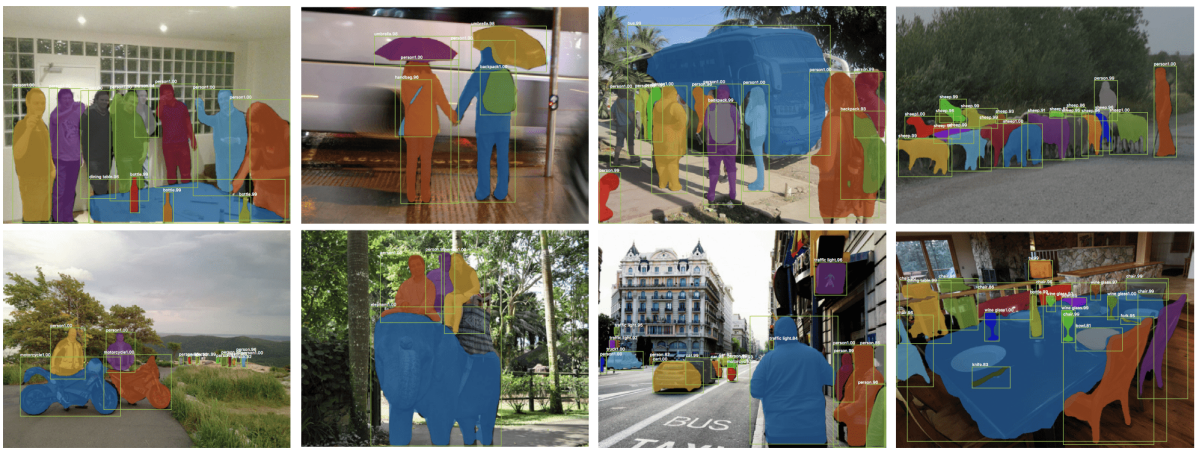


Figure 2.3: Example of Object Segmentation on the COCO Dataset [31] Taken from "Mask R-CNN".[21]

2.1.2. Convolutional Neural Network

Initially inspired by the visual system and models like Neocognitron (Fukushima, 1980) [15], which emphasized local connectivities and hierarchical transformations, CNNs gained

prominence through pioneering work by Yann LeCun and collaborators in the late 1980s[28]. This early research laid the groundwork for the development of CNNs, paving the way for their widespread adoption in various applications.

Due to the reduction in the number of parameters, CNNs can be applied to solve complex tasks that were impossible with classical artificial neural networks. This reduction is made possible by the unique structure and functioning methodology of CNNs. Unlike traditional neural networks, where each neuron in a hidden layer is connected to all neurons in the previous layer, CNNs employ a weight-sharing technique and a convolutional structure. This approach significantly decreases the number of parameters that need to be trained, resulting in more efficient processing of high-dimensional data such as images.

By applying convolutional filters on portions of the image, CNNs can effectively identify and extract the most relevant features. This feature extraction process enables CNNs to learn intricate patterns from data and generalize better to new examples. Consequently, CNNs excel in solving complex computer vision tasks such as object recognition, motion tracking, and medical image analysis. Through their capabilities, CNNs have transcended the limitations of traditional neural networks, ushering in a new era of possibilities in artificial intelligence and computer vision.

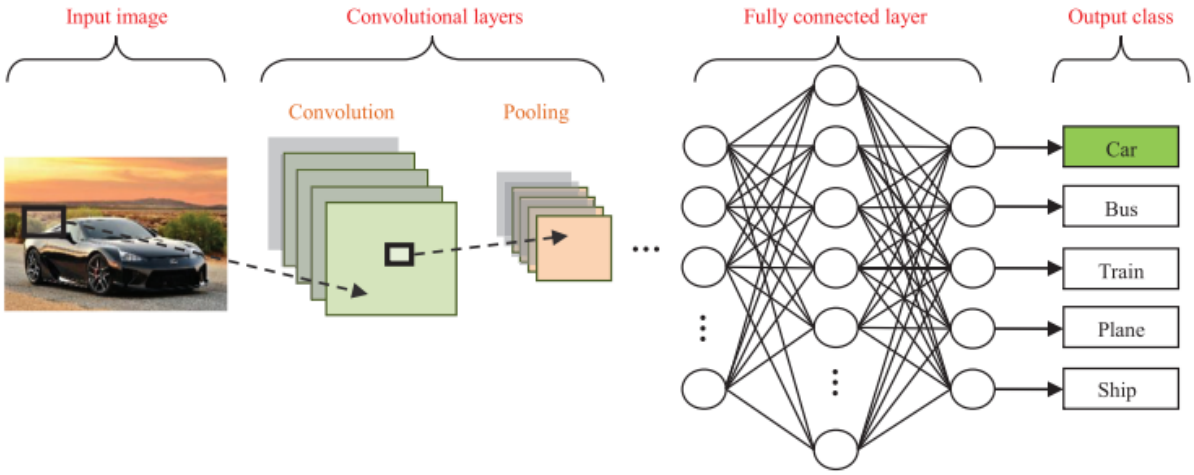


Figure 2.4: The pipeline of the general CNN architecture.[18]

A Convolutional Neural Network (CNN) consists of four primary types of neural layers: convolutional layers, pooling layers, activation layers and fully connected layers. Each of these layers serves a distinct role in processing input data. The figure above illustrates the architecture of a CNN designed for image classification in images. Each layer within the CNN transforms the input volume into an output volume of neuron activations. This

process continues through the layers until it culminates in the final fully connected layers, which ultimately map the input data to a one-dimensional feature vector.[54]

Convolution Operation The convolutional layers play a crucial role as feature extractors within Convolutional Neural Networks (CNNs), facilitating the learning of feature representations from input images. Within these layers, neurons are organized into feature maps, each neuron possessing a receptive field connected to neighboring neurons in the preceding layer via a set of trainable weights, often termed as a filter bank. This arrangement allows inputs to undergo convolution with learned weights, generating new feature maps, which are then subjected to a nonlinear activation function. It's notable that neurons within a feature map share equal weights, while different feature maps within the same convolutional layer possess distinct weights, enabling the extraction of multiple features at each location [29] .

Formally, the computation of the k-th output feature map Y_k can be expressed as:

$$Y_k = f(W_k * x) \tag{2.1}$$

where x denotes the input image, W_k represents the convolutional filter associated with the kth feature map, $*$ symbolizes the 2D convolutional operator and $f()$ denotes the nonlinear activation function [55]. The utilization of nonlinear activation functions is pivotal for extracting nonlinear features, with Rectified Linear Units (ReLUs) gaining popularity due to their effectiveness [39]. This shift towards ReLUs has spurred research into novel activation functions aimed at enhancing various aspects of CNN performance.

Convolutional layers in CNNs leverage diverse kernels to convolve both the entire image and intermediate feature maps, thereby generating a multitude of feature maps. This convolution operation offers several advantages: firstly, weight sharing within the same feature map reduces parameter count; secondly, local connectivity enables the learning of correlations among neighboring pixels; thirdly, it ensures invariance to the object's location[18]. Given these benefits, convolution has been adopted in lieu of fully connected layers in some instances to expedite learning .

So, a kernel is a compact 2D array whose values correspond to specific operations to be executed. Through basic matrix operations like multiplication and addition, a kernel is applied to an input image, resulting in an output with reduced dimensions, facilitating further processing. Here's an illustration of how kernels are utilized for tasks such as Gaussian blur (to soften the image), sharpening (to enhance edge definition), and edge detection.

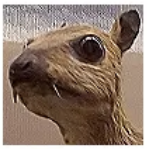
Original	Gaussian Blur	Sharpen	Edge Detection
$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$
			

Figure 2.5: Examples of kernel filters for CNN's.

The essential parameters for convolutional layers in a Convolutional Neural Network (CNN) include the kernel shape, the number of filters, and the stride, which define how the input convolution takes place.

Padding, such as "valid" and "same", plays a crucial role in shaping the output dimensions of convolutional layers and, consequently, the overall architecture and performance of the CNN. "Valid" padding, the most common type, indicates the absence of padding, meaning the filter is applied only where the input matrix and the filter fully overlap. This may lead to a reduction in output dimensions compared to the input, especially when the kernel is larger than the input.

On the other hand, "same" padding is designed to produce an output with the same dimensions as the original input by adding sufficient zero padding around the input borders. This is particularly useful when maintaining a constant image size across various convolutional layers. "Same" padding is often used in combination with odd-sized kernels to ensure the kernel's center is effectively positioned on the input's central element.

The choice of kernel size and the number of filters (or kernels) in a convolutional layer is equally important. Smaller kernels can capture finer details in the input, while larger kernels can capture broader and more global features. However, using larger kernels can significantly increase the model's parameter count and thus require more computational resources during training and inference.

The number of filters in a convolutional layer determines how many different types of features the layer's output will be able to capture. For example, if a layer has 32 filters, the output will contain 32 different feature maps, each representing a different combination of features extracted from the input. Therefore, the choice of kernel size and the number of filters must be carefully balanced to achieve a compromise between the model's complexity and its learning and generalization capabilities.

Pooling Layer Pooling layers, a fundamental component in Convolutional Neural Networks (CNNs), typically succeed convolutional layers and serve to decrease the dimensions of feature maps and network parameters. Much like convolutional layers, pooling layers exhibit translation invariance as they consider neighboring pixels during computations. The two most commonly employed pooling strategies are average pooling and max pooling [3, 28]. For instance, consider max pooling applied to 8x8 feature maps, resulting in output maps reduced to 4x4 dimensions, utilizing a max pooling operator of size 2x2 and stride 2. While both max pooling and average pooling have been extensively studied, research has shown that max pooling often leads to faster convergence, selects superior invariant features, and enhances generalization. Recent advancements in CNN architectures have predominantly utilized max pooling due to its efficacy in processing speed and feature selection capabilities [44].

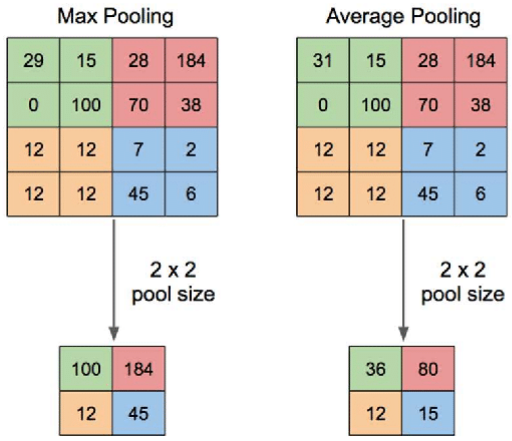


Figure 2.6: The operations of the pooling layers.

Pooling layers have garnered significant attention and are categorized into three well-known approaches, each serving distinct purposes. First, stochastic pooling addresses the sensitivity of max pooling to overfitting by introducing a stochastic procedure, enhancing generalization. Spatial pyramid pooling alleviates the limitation of fixed-size input images, offering a flexible solution for handling diverse scales, sizes, and aspect ratios. Finally, def-pooling tackles deformation challenges by learning the deformation constraints and geometric models of object parts, enriching the deep model's capabilities [18].

Activation Layer After the convolutional layer, an activation layer is applied to introduce non-linear transformations, enabling the model to capture intricate relationships within the data. Historically, activation functions like sigmoid and tanh were prevalent choices [49]. Sigmoid confines neuron outputs between 0 and 1, beneficial for tasks like bi-

nary classification. Tanh compresses outputs between -1 and 1, advantageous for scenarios with prevalent negative values, such as sentiment analysis.

Recently, the Rectified Linear Unit (ReLU) has gained prominence for several reasons. It boasts a simple definition and gradient, aiding in model training. Unlike sigmoid and tanh, ReLU helps alleviate the vanishing gradient problem in deep networks by maintaining a constant gradient for positive inputs. Moreover, ReLU promotes sparsity within the network's representation by deactivating neurons with negative inputs, while also exhibiting scale invariance [39]. However, ReLU has drawbacks, including non-differentiability at 0 and the issue of "Dying ReLU," where neurons become inactive for many inputs. Variants like Leaky ReLU have emerged to address these concerns by maintaining a small, non-zero gradient for negative inputs, thereby enhancing network performance [9].

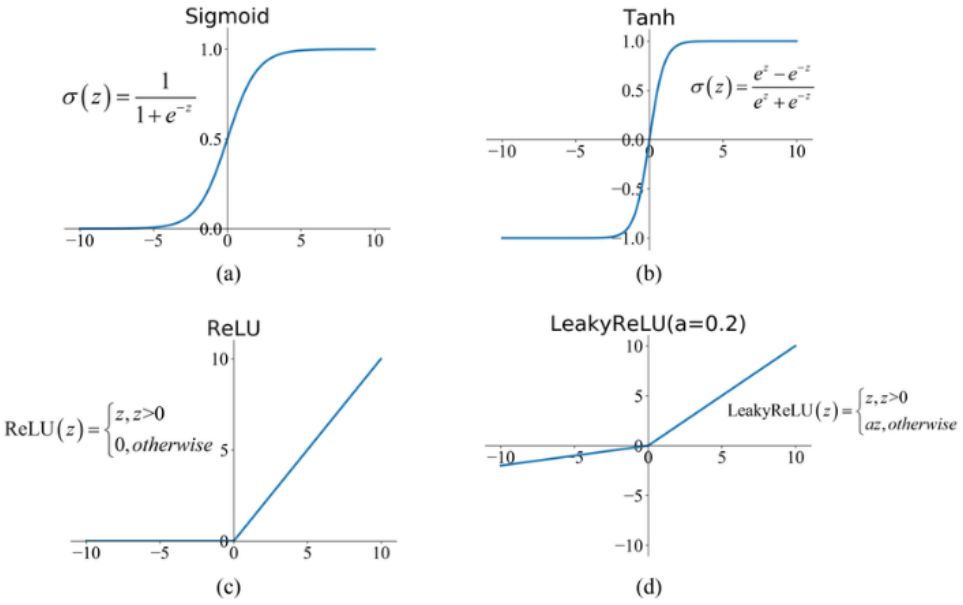


Figure 2.7: Some of the most popular activation functions.

Fully Connected Layer The fully connected layer, also known as the Dense layer, plays a crucial role in neural networks, including Convolutional Neural Networks (CNNs). It serves as a vital component in processing features extracted from earlier layers, combining them to perform predictions or classifications. Unlike convolutional layers, which establish connections within localized regions of the input, each neuron in a fully connected layer is linked to every neuron in the preceding layer.

This extensive interconnection enables the fully connected layer to discern intricate patterns and correlations within the data, proving particularly valuable for tasks such as image classification, object detection, and natural language processing. Neurons within

the fully connected layer assign weights to input features and subject them to an activation function, typically ReLU or softmax, to yield the final output.

However, the dense connectivity characteristic of fully connected layers presents certain drawbacks. They necessitate a substantial number of parameters, rendering them computationally demanding and susceptible to overfitting, especially when confronted with high-dimensional data. Additionally, fully connected layers fail to retain spatial information, which holds significance for tasks like image segmentation and object localization.

To address these challenges, fully connected layers are often followed by dropout layers during training, randomly deactivating some neurons to mitigate overfitting. Furthermore, in CNN architectures, fully connected layers are typically positioned towards the network's final layers, succeeding several convolutional and pooling operations, to distill high-level features before making predictions.

In conclusion, fully connected layers constitute indispensable elements of neural networks, facilitating the learning of complex patterns and facilitating predictions across diverse domains. However, their dense connectivity and computational complexity necessitate meticulous design and regularization to uphold optimal performance and generalization.

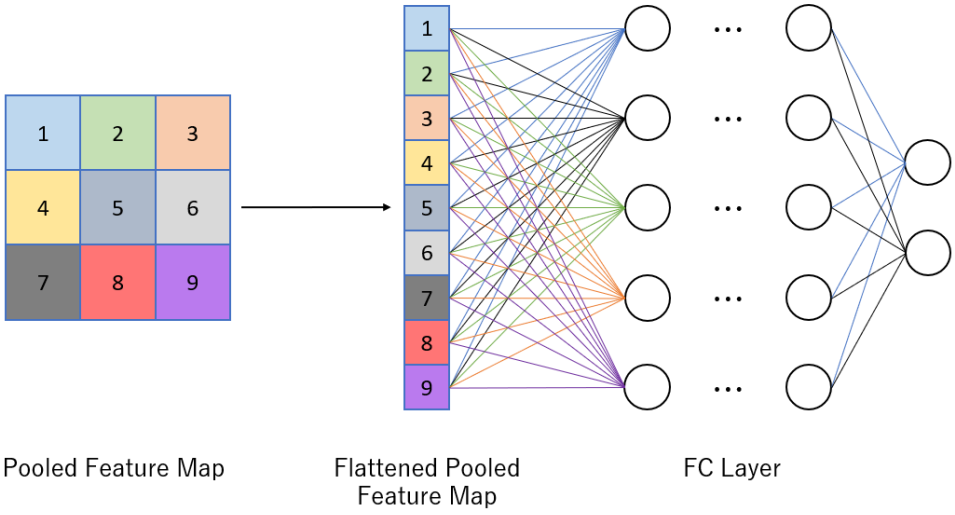


Figure 2.8: A pooled feature map being flattened and inputted into a fully connected (FC) network layer.

2.2. Learning Approaches

Machine learning encompasses a broad array of techniques and approaches. Among the many types of learning, some of the most common include:

- Supervised learning involves the training of a model using a dataset containing inputs along with their associated outputs. Throughout the training regimen, the model endeavors to comprehend the correlation between the inputs and outputs provided. Armed with this knowledge post-training, the model is capable of making predictions on fresh data by extrapolating from its learned patterns. [8].
- In weakly supervised learning the training data consists of only a fraction of labeled examples, leaving the majority unlabeled. Consequently, the model operates with incomplete information, relying on partial data to discern the underlying patterns and make predictions. [62].
- Self-learning is a method where unlabeled data is harnessed to enhance the performance of the model. During this process, the model leverages its predictions on unlabeled data to create additional labels, thereby refining its training set. This iterative cycle of self-improvement contributes to the gradual enhancement of the model's performance.
- Unsupervised learning involves training a model on data without explicit labels to uncover intrinsic patterns or structures within the dataset. This learning approach is frequently employed in tasks such as clustering, reducing the dimensionality of the dataset, and anomaly detection. [16].
- In reinforcement learning, an agent learns to interact within an environment to maximize its total reward over time. The agent makes decisions based on the current state of the environment and receives feedback through rewards or penalties. The agent's objective is to learn a policy that directs actions toward achieving maximum long-term reward. [25].
- Semi-supervised learning combines principles from both supervised and unsupervised learning methodologies. This technique involves training the model on a dataset comprising both labeled and unlabeled data points. By incorporating unlabeled data, the model can potentially improve its performance, particularly in scenarios where labeling all data is resource-intensive or unfeasible. [63].

These various learning methodologies provide versatile and robust strategies for training machine learning models, each possessing distinct advantages and limitations that necessitate careful consideration within the application context.

2.2.1. Supervised Object Detection

Object Detection, also termed object categorization, involves identifying and locating objects within images or videos based on predefined categories. It encompasses a broad range of natural categories in visual data, not limited to specific types like faces or vehicles. The process includes detecting objects, categorizing them, and providing detailed spatial information.

Spatial characteristics of objects are described using methods such as bounding boxes and enclosed boundaries. Bounding boxes are commonly preferred due to their simplicity and effectiveness [48].

Object detection faces challenges in accurately discerning and classifying objects amidst cluttered backgrounds and varying visual attributes. Unlike object classification, it involves determining spatial locations within images. Object detection is closely related to tasks like object recognition, semantic segmentation, and object instance segmentation, essential for advancing visual comprehension.

With the increasing use of social media and mobile devices, there's a growing demand for analyzing visual data. Efficient object detection is crucial, especially for mobile and wearable devices with computational and storage limitations. Efficiency hurdles arise from the need for precise localization and identification, escalating computational complexity. Scalability is another challenge, requiring detectors to handle new objects and scenarios adeptly. As manual annotation becomes impractical with increasing data volumes, weakly supervised learning strategies are being explored for training object detection models.s[34].

Methodologies in Object Detection:

- One-Stage Methods: One-stage methods in object detection emphasize real-time inference speed without compromising much on accuracy. These methods directly predict bounding boxes and class probabilities in a single pass through the neural network. Notable examples of one-stage methods include YOLO (You Only Look Once)[45], SSD (Single Shot MultiBox Detector) [35], and RetinaNet[32]. YOLO, in particular, introduced the concept of dividing the input image into a grid and predicting bounding boxes and class probabilities for each grid cell, thus achieving remarkable speed in detection.

- Two-Stage Methods: In contrast, two-stage methods prioritize detection accuracy over speed. These methods typically consist of a region proposal network (RPN) followed by a bounding box refinement network. The region proposal network generates potential object bounding boxes, which are subsequently refined and classi-

fied in the second stage. Prominent examples of two-stage methods include Faster R-CNN[46], Mask R-CNN[22], and Cascade R-CNN[5]. These methods excel in accurately localizing objects with precise boundaries and category labels.

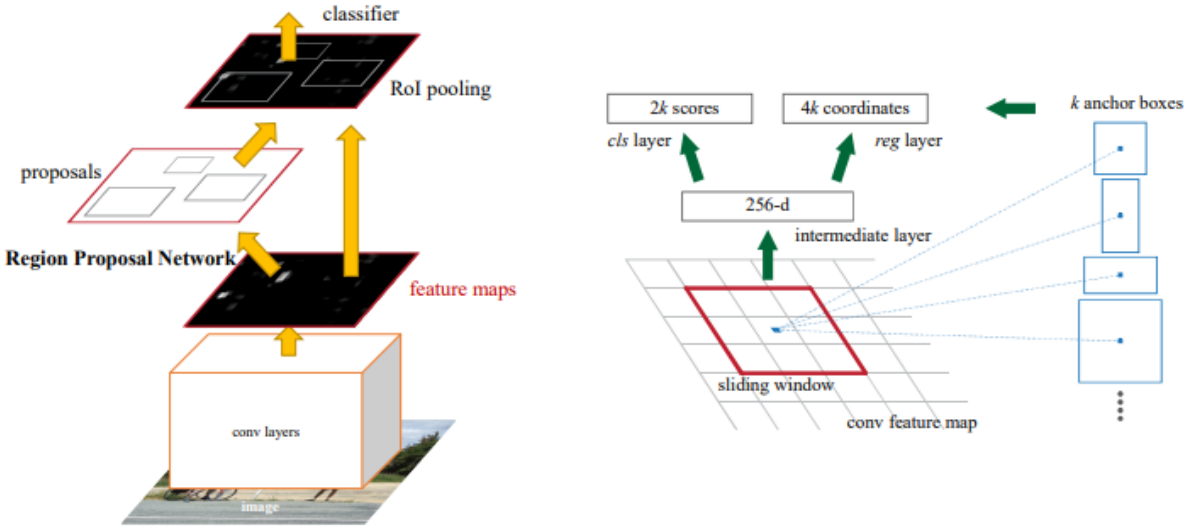


Figure 2.9: Left: framework of the faster R-CNN for object detection with a single network. Right: Region Proposal Network (RPN). [46].

2.2.2. Supervised Object Segmentation

Supervised object segmentation plays a critical role in computer vision, leveraging annotated training data to direct the segmentation of images. In this technique, every pixel in the image is linked to a predefined class or category, enabling the machine learning algorithm to differentiate and classify pixels based on their visual properties. Accurate annotation of the training data is vital for this form of segmentation, wherein each pixel in the image is manually labeled with its respective class. After model training, it can automatically partition new images, assigning each pixel a class predicted by the model. [38]

Recent advancements in supervised segmentation have undergone rapid progression, primarily attributable to the advancements in deep learning and convolutional neural networks (CNNs). These technological innovations have facilitated the creation of progressively sophisticated and precise models for image segmentation. These models exhibit the capability to discern and differentiate a diverse array of objects and visual elements with exceptional accuracy.

Various types of approaches to supervised segmentation include:

- Region-based methods: these methods involve segmenting the image into homogeneous regions, each of which may be associated with an annotated class. These regions can be obtained using techniques such as region growing, region splitting and merging, or watershed segmentation. Once the regions are extracted, features are extracted from the regions and used to train a classifier that can be used to predict the classes of regions in new images. For example, the paper "Efficient Graph-Based Image Segmentation" by P. F. Felzenszwalb and D. P. Huttenlocher presents a graph-based segmentation algorithm that merges regions based on a criterion derived from the difference in intensity between neighboring pixels and the size of the regions[12].
- CNN-based methods: These methods leverage Convolutional Neural Networks (CNNs) to perform image segmentation. CNNs can learn directly from images and generate pixel-wise segmentation maps. There are various CNN architectures specifically designed for segmentation, such as U-Net[47], Fully Convolutional Network (FCN) [36], Mask R-CNN [22], and others. These models are trained using annotated images, where both the inputs (images) and outputs (segmentation maps) are provided during the training process. For example, the paper "U-Net: Convolutional Networks for Biomedical Image Segmentation" by O. Ronneberger, P. Fischer, and T. Brox proposes the U-Net architecture that is specifically designed for biomedical image segmentation tasks, featuring a contracting path for capturing context and a symmetric expanding path for precise localization[47].

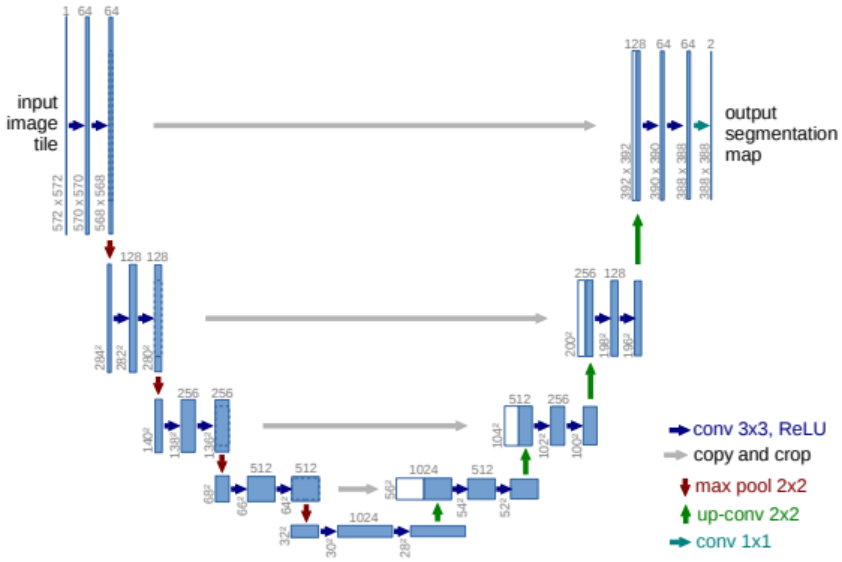


Figure 2.10: U-net architecture [47].

In summary, supervised segmentation is a critical technique in computer vision, allowing for the extraction of detailed information from images using annotated training data. Recent advancements in deep learning have led to significant improvements in the accuracy and reliability of supervised segmentation models, opening up new opportunities for applications in various fields such as medicine, robotics, autonomous vehicles, and more [19].

2.2.3. Weakly-Supervised Localization

Weakly-Supervised Localization (WSL) is a computer vision technique aimed at identifying the position of objects of interest in an image using only minimal supervision. Instead of requiring detailed annotations (such as precise bounding boxes) to train a localization model, WSL relies on more general or less costly annotations to obtain, such as class labels or annotations of entire images.

There are several main approaches to weakly supervised localization:

- Class Activation Mapping (CAM): CAM is a technique that leverages the power of convolutional neural networks (CNNs) to generate class activation maps, indicating the most relevant regions of the image for predicting a particular class. These maps can be used to localize the object of interest without requiring precise localization labels.
- Multiple Instance Learning (MIL): MIL is a machine learning paradigm where the supervision label is assigned to a set of instances rather than individual instances. In the context of weakly supervised localization, MIL can be used to train a model to recognize the presence of an object in an image without requiring precise bounding box labels.
- Self-training: This approach leverages the model's ability to self-generate weakly supervised labels. Initially, a model is trained on a set of weakly labeled or unlabeled data, and then this model is used to make predictions on a larger set of data. The generated predictions are used to label additional data, which is then incorporated into the training set for a new training cycle. This process can be iterated to progressively improve localization performance.

Class Activation Mapping (CAM)) Weakly supervised localization with the Class Activation Mapping (CAM) approach utilizes convolutional neural networks (CNNs) to generate class activation maps (CAMs). These CAMs highlight the most discriminative regions of an image for predicting a particular class. These localization properties of

CNNs have been further studied in other works such as [40, 56]. The process begins with training a CNN on a dataset that contains image-level labels indicating the presence of objects but lacks precise localization information. Once the CNN is trained, it generates CAMs by weighting the feature maps of its last convolutional layer by the corresponding class scores and aggregating them to produce a heatmap for each class. These heatmaps effectively indicate the regions of the image that are most relevant for predicting the target class.

In the application phase, these CAMs serve as a mechanism for weakly supervised localization. By examining the CAMs, analysts can identify regions of the image that are most likely to contain the object of interest, even in the absence of precise bounding box annotations. Regions with high activation values correspond to areas of the image that are particularly relevant for predicting the presence of the target class. However, it's important to note that CAMs may produce inaccurate or ambiguous localizations, especially for objects with complex backgrounds or overlapping instances.

Despite these limitations, weakly supervised localization with CAMs offers an efficient means to localize objects in images using only image-level labels for supervision. This approach has found applications in various domains, including scene understanding, content-based image retrieval, and weakly supervised object detection. Ongoing research continues to explore ways to enhance the accuracy and robustness of weakly supervised localization methods based on CAMs.

The paper "Learning deep features for discriminative localization" by Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba is a pioneering contribution in the field of computer vision[60]. This work was one of the first to introduce the concept of Class Activation Mapping (CAM) for weakly supervised object localization.

In this paper, the authors present an innovative method for generating class activation maps, which highlight the most relevant regions of the image for predicting a specific class. The approach relies on leveraging information from the last convolutional layer of a convolutional neural network (CNN) to obtain a class activation map.

The proposed formula for calculating CAMs is simple and effective:

$$L^c_{\text{CAM}} = \sum_k w^c_k \cdot A^k$$

Where:

- L^c_{CAM} represents the class activation score for class c.

- w_k^c are the weights associated with the channels of the last convolutional layer for class c.
- A^k are the activation maps of the last convolutional layer.

Essentially, this formula computes the class activation score for each class c by summing the product of the weights of the channels for that class with the corresponding activation maps of the channels.

The class activation maps thus obtained are then used to generate object localization maps, where regions with higher activation scores indicate the most relevant areas of the image for the class of interest.

This paper has had a significant impact in the field of computer vision, as it introduced an effective and interpretable method for weakly supervised object localization, paving the way for numerous subsequent developments and applications based on CAMs.

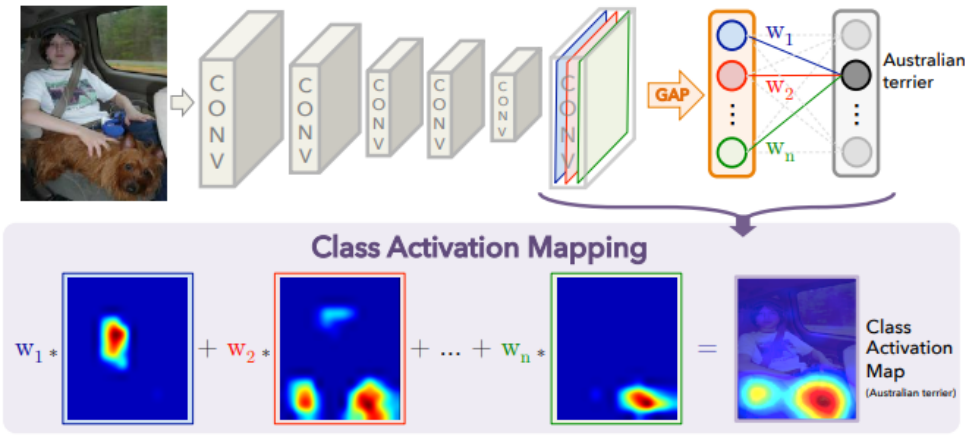


Figure 2.11: Class Activation Mapping involves mapping the predicted class score back to the preceding convolutional layer to generate class activation maps (CAMs)[60]

However, there are several issues associated with this approach. One issue is the imprecise localization of objects. CAMs may struggle to accurately localize objects within images, often identifying generic regions rather than exact object contours. This is because CNNs were not originally designed for object localization but rather for extracting discriminative features.

Another limitation is the restriction to a single fully connected layer. CAMs typically require the use of only one fully connected layer at the end of the CNN. This limits the model's flexibility, as sacrificing model complexity may be necessary to integrate CAM generation.

Additionally, problems can arise with techniques such as Global Average Pooling (GAP) or Global Max Pooling (GMP). CAMs often employ global pooling techniques to reduce dimensionality before the fully connected layer. However, these techniques may lead to a loss of detailed spatial information, which could negatively impact CAM quality and object localization accuracy.

CAMs may also exhibit sensitivity to background classes, highlighting irrelevant regions within images. This can be problematic, particularly in scenarios where images contain many background elements unrelated to the class of interest.

Grad-CAM++ Grad-CAM++ represents an evolution of Class Activation Maps (CAM), designed to overcome some of the limitations encountered in CAM generation. Introduced by Chattopadhyay et al. in 2018, it introduces enhancements to improve object localization and the overall quality of activation maps[7].

One key enhancement in Grad-CAM++ involves the utilization of higher-order gradient weights. By capturing more complex relationships between CNN feature maps and the output gradient concerning those feature maps, Grad-CAM++ achieves greater sensitivity to low-level features and refined object localization.

Moreover, it integrates multiple layers of the network to generate activation maps, as opposed to the original CAM which relies solely on a single layer. This approach enables the capture of a wider range of spatial and semantic information, thereby enhancing the overall quality of activation maps.

Additionally, Grad-CAM++ employs weighted pooling based on gradient weights, considering higher-order gradient weights to aggregate information from feature maps. This helps mitigate the loss of spatial details during activation map generation.

Lastly, to ensure greater stability and consistency in activation map generation, normalization of the gradient weights is performed before using them to weigh the feature maps.

In summary, Grad-CAM++ aims to improve the quality and accuracy of activation maps compared to original CAMs by addressing issues such as imprecise object localization and sensitivity to background classes, making it a more powerful and reliable technique for visualizing the decisions of convolutional neural networks.

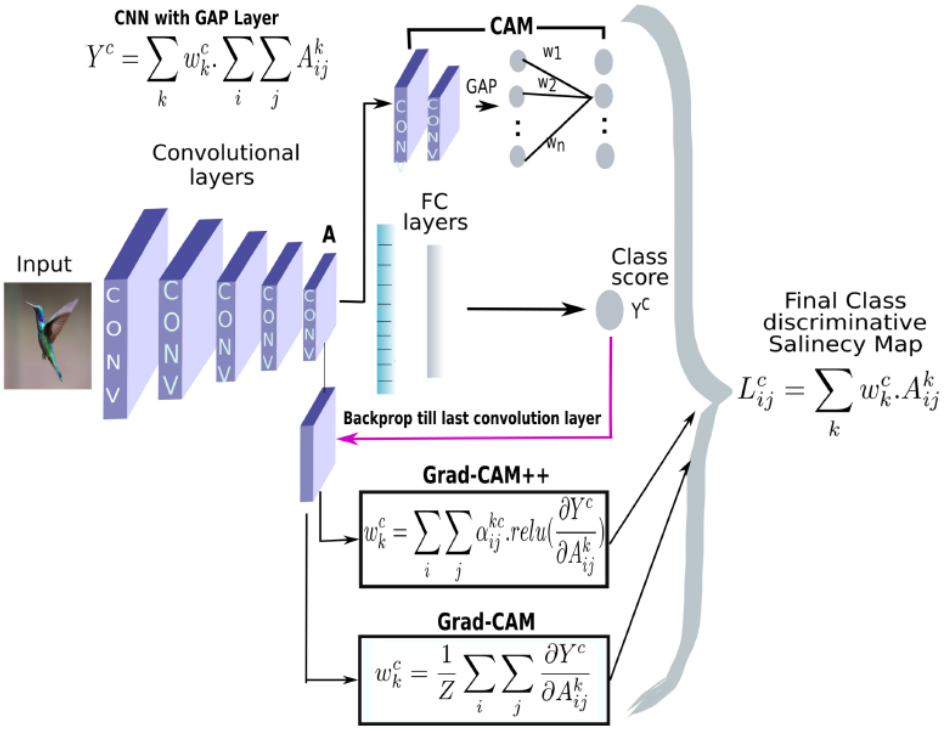


Figure 2.12: Grad-CAM++[7]

Multi-Instance Learning (MIL) Multiple instance learning (MIL) represents a form of weakly supervised learning where data instances are grouped into sets called bags. Instead of assigning labels to individual instances, labels are attributed to entire bags. Consequently, the objective in MIL is to comprehend a concept based on the labels associated with these collections of instances.

In MIL, given an entire starting image, its patches are the instances, and a bag is a collection of such patches that can be all or a subset of the set of possible patches from the starting image. The label associated with the starting image is then assigned to the bag of patches. So, the key idea is that instead of labeling individual patches, MIL labels collections of patches (bags), and the label of a bag is determined based on the presence or absence of positive instances within it. If at least one patch within the bag is labeled positive, the entire bag is labeled positive; otherwise, it is labeled negative[4].

Deep multiple instance learning addresses the challenges of MIL by training a deep MIL model using the log-likelihood function. In this context, the bag label follows a Bernoulli distribution with a parameter θ(X) ranging from 0 to 1, which denotes the probability of Y = 1 given the bag of instances X.

Multiple Instance Learning (MIL) framework consists of four key steps:

- A patch selection method dictates the mechanisms of how the patches are selected from the entire collection of the possible patches.
- A transformation function $f(\mathrm{x})$ converting a image into a lower representation. $f(\mathrm{x})$ is basically a computer vision model that converts a patch into a patch-level feature representation, which can be a multi-dimensional vector (lower dimensions than the patch) or a score (1-dimensional vector).
- A pooling operation that aggregates the patch-level feature representations of all the proposed patches into a single feature representation.
- A predictor $g(\mathrm{X})$ that converts the single feature representation into the final score for the bag.

Both the transformations f and g can be implemented using neural networks, offering flexibility and allowing end-to-end training via backpropagation. The only requirement is that the MIL pooling operation must be differentiable.

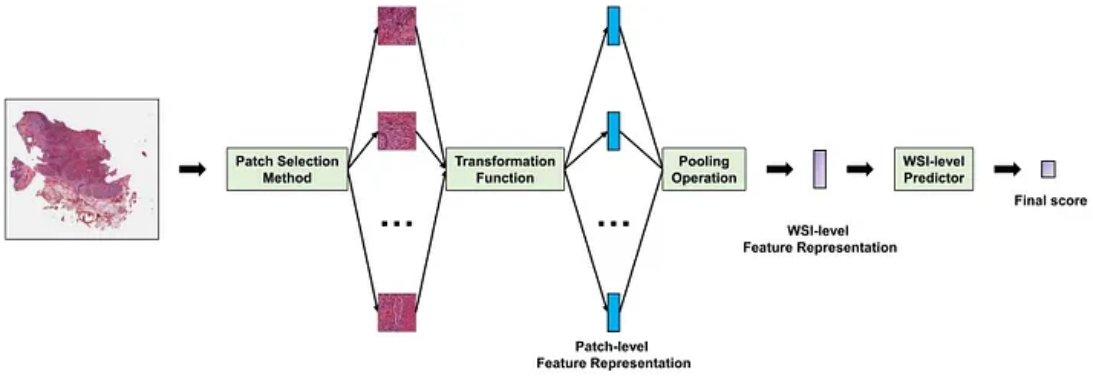


Figure 2.13: Multiple Instance Learning (MIL) framework

In multi-instance learning (MIL), the extraction of patches from the entire image plays a crucial role. Various techniques are employed to achieve this:

- Random Selection: This basic approach involves randomly choosing patches without considering specific criteria. While it may reduce computational costs, it could inadvertently generate bags without positive instances, leading to mislabeling issues.
- Random Selection within Regions of Interest (ROI): This method involves randomly selecting patches within manually annotated Regions of Interest (ROIs).
- Grid-Based Strategy: This technique divides the image into a regular grid and extracts patches from each grid cell. While simple, it may not effectively capture objects of different sizes or shapes.

- Key Point Detection and Descriptors: Algorithms like SIFT or SURF are utilized to detect key points in the image and extract patches around these points. This method is beneficial for capturing regions with distinctive features.
- Convolutional Neural Network (CNN)-Based Approach: Employ pre-trained CNNs to automatically extract relevant features from the image, using methods such as sliding windows or pooling algorithms.
- Hierarchical Selection: Utilize a multi-level approach to patch extraction, capturing features at different levels of abstraction or granularity within the image. May start with larger, more general patches and progress to smaller patches to capture finer details.

Another important key step is the pooling operation, which aims to aggregate the various patch-level feature representations into a single bag-level feature representation. These operations can be mainly divided into 2 types:

- Static Pooling: What distinguishes this category is its stability throughout the training process; these operations remain constant. Examples include max pooling and average pooling.
- Adaptive Pooling: Methods in this category dynamically adjust the contribution of patches. Notable examples include Attention-based pooling [24] and Hopfield pooling [43], both of which fall under trainable pooling.

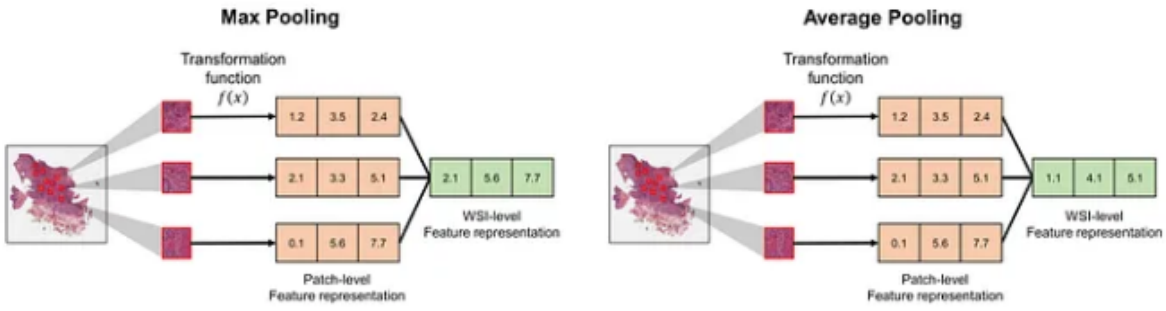


Figure 2.14: Max pooling and average pooling on the patch-level feature representations.

Among the most common issues encountered in Multi-Instance Learning (MIL), several challenges arise, including [6]:

- Label Ambiguity: In MIL, labels are assigned to bags of instances rather than individual instances. This can lead to label ambiguity as it's unclear which instances within a bag actually contribute to the positive or negative label of the bag itself.

- Presence of Noisy Negative Instances: Negative instance bags may contain "noisy" positive instances—instances that shouldn't be considered positive examples but are erroneously included in the bag. This can make it difficult for MIL algorithms to correctly discriminate between positive and negative instances.
- Partial or Missing Information: Instance bags may contain only partial or incomplete information about a particular concept or class, making it challenging for MIL algorithms to learn accurate and generalizable models.
- High Dimensionality: In many MIL applications such as object recognition in images or anomaly detection in medical data, the number of instances and data dimensions can be very high. This can lead to high-dimensional problems and may require dimensionality reduction techniques or scalable learning algorithms to effectively address the issue.
- Bag Imbalance: Instance bags can vary considerably in terms of the number of instances and the quality of information. Some bags may contain many informative instances, while others may contain only a few or irrelevant instances. This imbalance can affect the ability of MIL algorithms to learn accurate and representative models.
- Computational Complexity: Some MIL algorithms can be computationally intensive, especially when working with large volumes of data or problems with high dimensionality. This can limit the scalability of the algorithms and require significant computational resources for model training and processing.

2.2.4. Weakly-Supervised Segmentation

Weakly supervised segmentation is an approach where the machine learning model is trained with less precise or more economical supervision labels compared to fully annotated segmentation. This type of supervision may require less human effort for data collection but might result in lower accuracy compared to fully supervised segmentation.

There are several main approaches to Weakly-Supervised Segmentation, the three most popular approches are:

- CAM-based (Class Activation Map-based): This technique uses Class Activation Maps (CAM) to identify regions relevant to a particular class in the image. CAMs are generated from convolutional neural networks and can be used to guide segmentation[41].
- MIL-based (Multiple Instance Learning): Multiple Instance Learning (MIL) is a su-

pervised learning paradigm where training labels are assigned to a "bag" of instances rather than individual instances. In segmentation, this may mean that labels are assigned to regions (instances) rather than individual pixels.

- Segments or Superpixels based on their similarity within or between images: In this approach, the image is divided into segments or superpixels, which are homogeneous regions based on certain features (such as color, texture, etc.). These segments or superpixels are then used to guide the segmentation process, for example, by using similarity measures between them to aggregate them into larger regions or differentiate regions of interest from the background [27].

Moreover, there are frameworks available that can be categorized into one-stage and two-stage approaches:

- One-stage segmentation: In this approach, the model directly performs segmentation from the input image without a preliminary phase of feature extraction or region proposal generation. This approach is known for its efficiency in terms of processing time but may have lower accuracy compared to the two-stage approach[58].
- Two-stage segmentation: Here, the segmentation process occurs in two distinct stages: first, region proposals that might contain objects of interest are generated, and then these proposals are further processed to produce final segmentations. This approach tends to have higher accuracy but can be more computationally expensive[2, 26].

In summary, weakly supervised segmentation can utilize various approaches, including one or two-stage, CAM-based, MIL-based, or segments/superpixels based on their similarity, to identify regions of interest in an image using less expensive or more economical supervision labels.

2.3. Remote Sensing Imagery

Remote sensing images (RSIs) present unique challenges for deep learning applications, differing significantly from natural images. While natural images typically showcase prominent objects against simple backgrounds, RSIs exhibit complex spatial characteristics and diverse content. In literature, various papers have addressed challenges related to working with remote sensing imagery datasets [17, 30, 61]. Below, we explore the distinctive aspects of RSIs and the specific challenges they pose for deep learning algorithms, essential for developing effective models tailored to remote sensing applications.

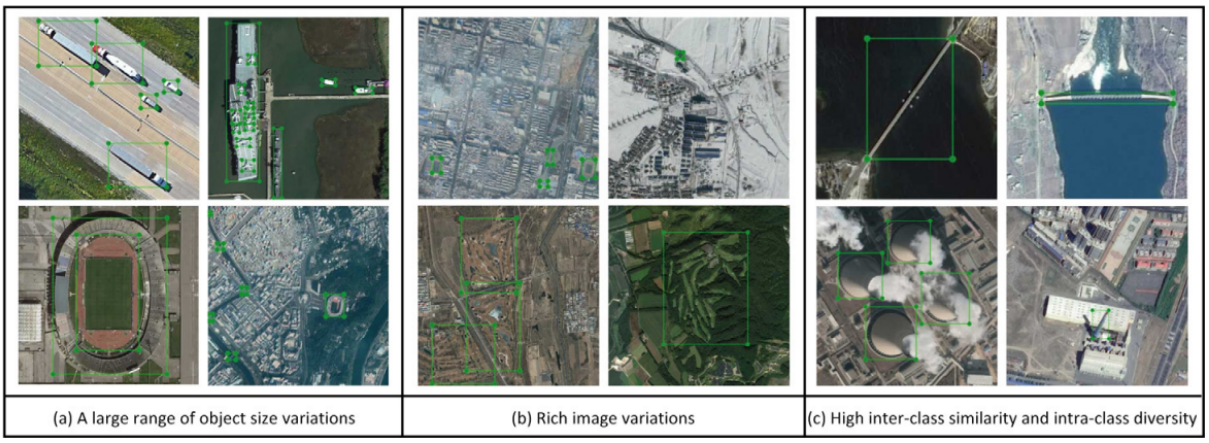


Figure 2.15: Some of the challenges on remote sensing images from the DIOR data set. [30]

1. Object Size and Spatial Distribution: RSIs often feature small object instances scattered across images, contrasting with the presence of a few large objects in natural images. Deep learning models for RSIs must excel in object localization and detection within larger image contexts.

2. Background Complexity: RSIs frequently depict backgrounds with high complexity, where multiple ground objects coexist. Extracting relevant information from objects amidst such diversity and overlap demands models capable of robustly distinguishing and segmenting objects from background clutter.

3. Object Size and Density: RSIs exhibit a wide range of object sizes, from extremely small and dense, like ships or vehicles, to expansive, as seen in agricultural fields. Effective deep learning models for RSIs must handle multi-scale object variations adeptly.

4. Object Orientation: Unlike natural images, objects in RSIs can assume arbitrary orientations, necessitating models capable of detecting and classifying objects regardless of orientation.

5. Intra-Class Diversity and Inter-Class Similarity: RSIs often feature high intra-class diversity and significant inter-class similarity. Deep learning models need to learn discriminative representations of objects and manage complex relationships between different classes effectively.

6. Spatial Information: RSIs capture information about land cover, such as building roofs or vegetation, whereas natural images convey object profiles. Deep learning models for RSIs must leverage spatial information effectively to make accurate

predictions.

Addressing these challenges requires specific approaches in deep learning model design, data management, and feature engineering tailored to RSIs. Integrating these considerations with the aforementioned challenges can foster the development of robust and effective deep learning models for remote sensing image analysis.

Below, we present some of the most renowned datasets in the field of remote sensing imagery, each characterized by specific challenges and peculiarities. These datasets are widely used for image analysis and the implementation of deep learning algorithms. However, it is important to note that each of these datasets is affected by particular issues that may influence the analysis process and model training. Let's now examine some of the most famous datasets, highlighting the challenges they pose and the papers that have utilized them to address such issues.

- ISPRS Potsdam and Vaihingen Datasets: These datasets contain high-resolution aerial images of the cities of Potsdam and Vaihingen in Germany. They are extensively used for semantic segmentation tasks and other advanced image analyses.
- SpaceNet Datasets: SpaceNet offers high-resolution satellite images of various cities and geographic regions. These datasets are employed in competitions and challenges to foster the development of advanced satellite image analysis algorithms [10].
- UC Merced Land Use Dataset: This dataset comprises multispectral satellite images representing 21 land use categories, captured over Merced County, California. It serves as a reference dataset for land use image classification tasks[57].
- EuroSAT Dataset: EuroSAT consists of multispectral satellite images covering 10 land cover categories, captured by Sentinel-2 satellites. It is a valuable resource for land cover image classification tasks [23].

These datasets provide valuable resources for researchers and practitioners in the field of remote sensing, despite the challenges they may present. Researchers have leveraged these datasets to develop and evaluate innovative solutions for various remote sensing applications.

3 | Proposed Solution

3.1. Dataset

To carry out this study, aimed at developing models capable of locating and identifying waste from satellite images in a weakly supervised manner, using only image labels (indicating whether waste is present in the image or not without providing any information about its location), we utilized the AerialWaste dataset[53]. This dataset collects satellite images of the Lombardia region annotated by ARPA agency, the environmental monitoring agency of the Lombardia region in Italy. The dataset was constructed for the specific task of discovering illegal landfills, initially to identify the presence or absence of waste in an area and extract its location, but subsequently expanding to also identify the type of waste.

There are different versions of the AerialWaste dataset due to updates that increase the number of images. For this study, we used version 2.0, which, unlike the first version comprising 10,434 RGB remote sensing images, includes an additional 543 images, totaling 10,977 RGB satellite images. These images were divided into 2 partitions: the training set and the testing set, to enable accurate and aligned comparisons among various studies on this dataset. Specifically, the training set contains 8,372 images out of the initial 10,977 images in the entire dataset, while the remaining 2,605 images are included in the testing set.

The satellite images available do not all come from the same source; rather, we can identify three different sources: AGEA Ortophotos, WorldView-3, and GoogleEarth. Depending on the source, the images have different sizes and Ground Sampling Distances (GSD), meaning different pixel resolutions based on the type of source from which the images originate.

Specifically, images from the Italian Agriculture Development Agency (AGEA) have a spatial resolution of approximately 20 cm Ground Sample Distance (GSD) and dimensions of 1,050×1,050 pixels. WorldView3 provides high-resolution images from a commercial satellite with a spatial resolution of around 30 cm GSD and dimensions of 700x700 pixels.

Additionally, images from Google Earth, obtained via the Google API, start with an initial spatial resolution of about 50 cm GSD. However, after post-processing by the API, the resolution improves to 21 cm GSD, with dimensions of 1000x1000 pixels.

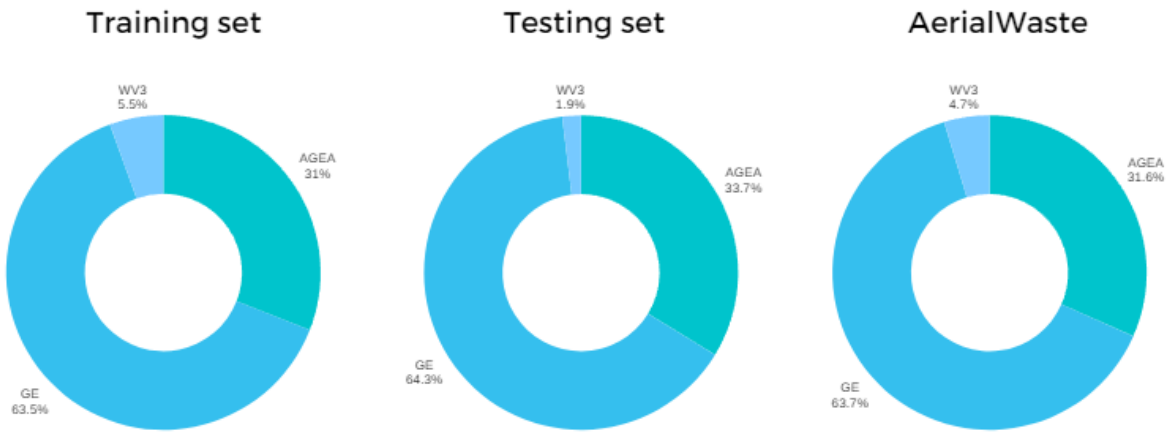


Figure 3.1: Distribution of the different sources for each partition of the dataset.

As we can observe from Figure 3.1, the distribution of sources differs within each partition (training, testing, dataset), but efforts are made to ensure that the distribution of these sources for each partition is similar, aiming to guarantee accurate evaluation during testing. It is evident that the most widely represented source is GoogleEarth with its 21 cm/pixel resolution, followed by AGEA with its 20 cm/pixel, which is the highest resolution available to us, and finally WorldView3 with its 30 cm/pixel.

As mentioned earlier, the AerialWaste dataset was initially created to determine whether waste is present or absent in a specific area. Thus, the primary annotation for each image in the dataset is the binary label, indicating a positive label for the presence of waste and a negative label for its absence. It's important to note that a positive label does not necessarily mean a single waste landfill; there could be multiple waste sites in the area. Conversely, a negative label confirms the absence of waste in the designated area.

As depicted in Figure 3.6, the distribution of positive and negative images is unequal, with a ratio of 1 to 2. This means that the number of negative examples is twice that of positive examples, a ratio consistent across the entire dataset and within each partition, including the training and testing sets.

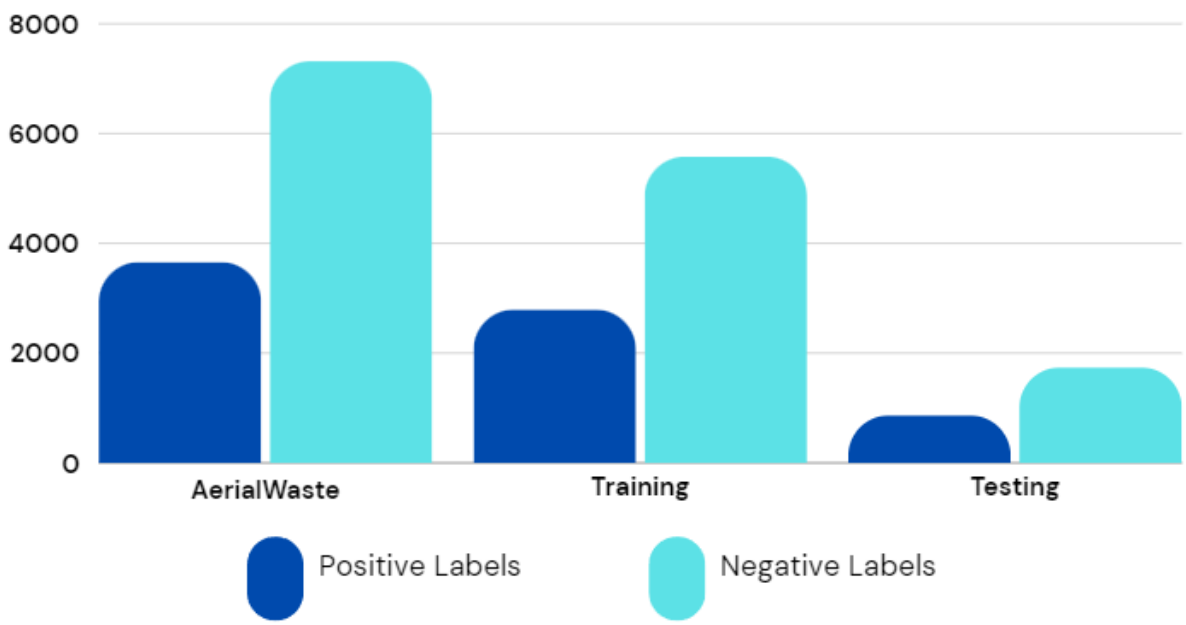


Figure 3.2: Distribution of the binary labels in the different partitions.

Specifically, in the training set, there are 5,579 negative images and 2,793 positive images, while in the testing set, there are 1,739 negative images and 866 positive images. In total, there are 7,318 negative images and 3,659 positive images.

Furthermore, some positive images in the dataset may have associated metadata, including but not limited to:

- Evidence: Indicates how many indicators of non-compliance with waste disposal regulations are detected by the analyst.
- Severity: Indicates the perceived severity of the non-compliance, expressed on a scale from 0 (low) to 3 (high).
- Category of the Area: Describes the category of the area where the waste is found, such as a production plant or agricultural area.

Subsequently, efforts were made to expand the tasks that could be performed with AerialWaste. In addition to identifying the presence or absence of waste in an area, the goal was to also identify the type of waste present. Therefore, a subset of images was annotated with the presence or absence of a specific type of waste in the image.

In total, within the AerialWaste dataset, there are 890 positive images annotated with the type of waste they contain, with 722 of these images present in the training set and the remaining 168 in the testing set.

It's possible to identify a total of 22 possible categories that can be assigned to a positive image containing waste. These categories can be further divided into two macro-categories:

- Waste Types: These categories represent the different types of objects that can be found in a landfill. The categories include rubble/excavated earth and rocks, bulky items, firewood, scrap, plastic, vehicles, tires, domestic appliances, paper, sludge-zootechnical waste-manure, stone/marble processing waste, asphalt milling, corrugated sheets (presumed asbestos-cement), glass, and foundry waste.
- Storage Modes: These categories pertain to the type of container or mode used to store waste. The categories include heaps not delimited, containers, big bags, pallets, delimited heaps (by barriers/walls/etc), cisterns, and drums bins.

	Aerialwaste	Training	Testing
Rubble	380	314	66
Bulky items	361	317	44
Fire Wood	201	163	38
Scrap	212	185	27
Plastic	157	133	24
Vehicles	70	44	26
Tires	52	39	13
Domestic appliances	26	21	5
Paper	33	28	5
Sludge-Zootechnical waste	23	19	4
Foundry waste	10	9	1
Stone/marble	16	15	1
Asphalt milling	12	9	3
Corrugated sheets	15	14	1
Glass	9	7	2

Table 3.1: The distribution of the waste types.

	Aerialwaste	Training	Testing
Heaps not delimited	584	491	93
Full container	169	115	54
Big bags	72	53	19
Full pallets	76	69	7
Delimited heaps	84	53	31
Cisterns	41	32	9
Drums bins	20	18	2

Table 3.2: The distribution of the storage modes.

Upon analyzing Tables 3.2 and 3.1, a significant issue becomes apparent: the imbalance problem, where certain waste type categories have a sufficiently high number of annotations to effectively train models. For instance, rubble exhibits 380 total annotations, with 314 for the training set and 66 for testing, while bulky items show 361 total annotations, with 317 for training and 44 for testing. However, there are categories with very few labels, such as asphalt milling with only 12 total annotations and glass with just 9. The well-known data hunger in deep learning models makes it challenging to train on limited data. Consequently, the scarcity of data for some categories currently prevents model training for identifying specific types.

Besides the imbalance problem, the AerialWaste dataset also presents challenges discussed in the Background section regarding satellite imagery. These challenges include complex background scenarios ranging from urban and industrial areas to rural landscapes. Additionally, waste items within the same category may vary significantly in size and appearance despite belonging to the same waste type.

Lastly, another crucial issue is the simultaneous occurrence of multiple types of waste within a single image. When certain waste types consistently appear together in images but rarely appear individually, it can undermine the classifier's ability to effectively discriminate between them. This phenomenon occurs because the network may predict the presence or absence of all co-occurring waste types without learning the distinctive features necessary to differentiate between them. Regrettably, this situation is frequently encountered in the dataset being analyzed.

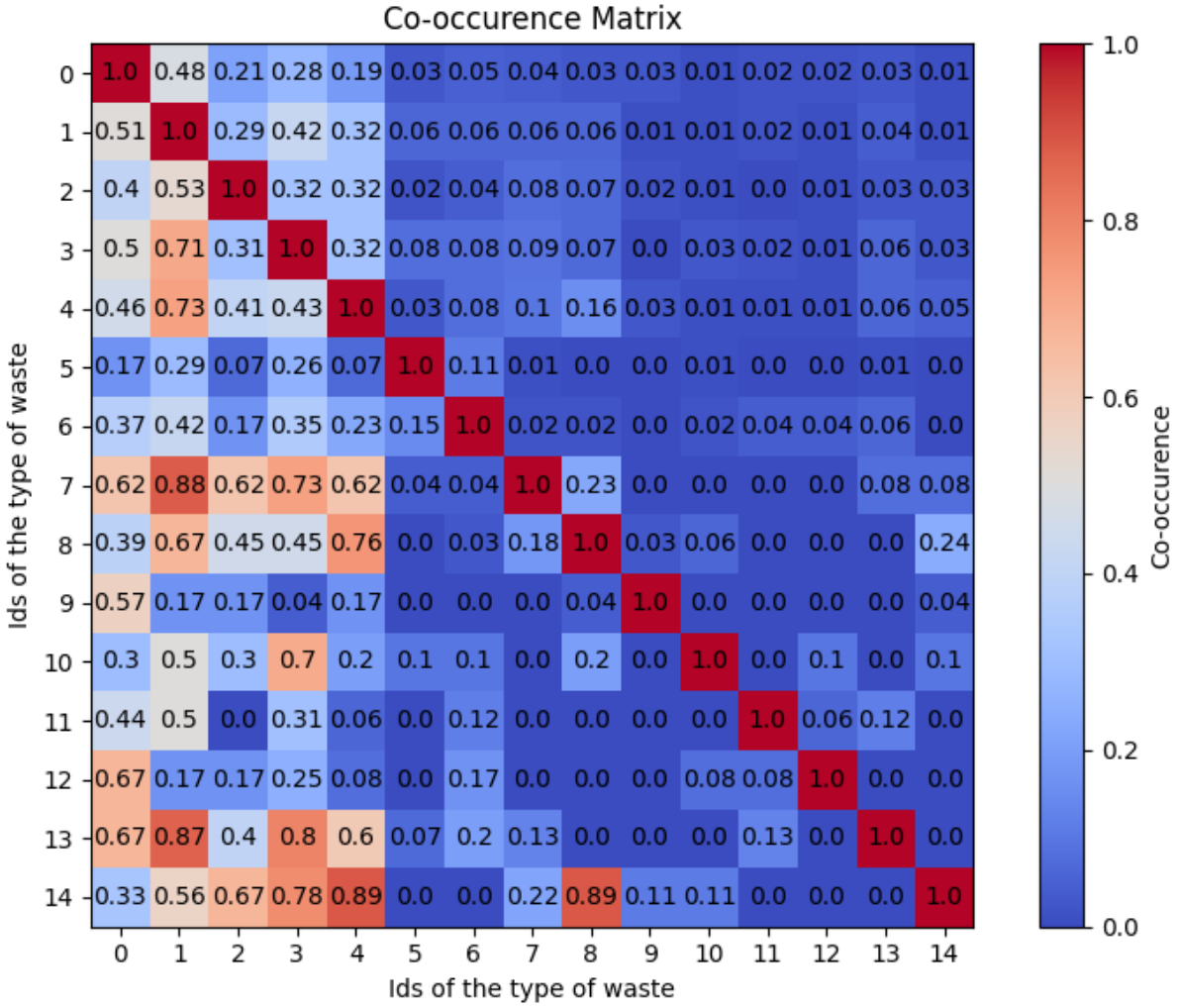


Figure 3.3: The co-occurrence matrix illustrates the relationships between different waste types within the dataset. Each cell represents the percentage of images where a particular type of waste is present alongside another specific waste type, across all images where it appears.

As can be observed from Figure 3.3, which presents the co-occurrence matrix, the issue we just discussed becomes evident. For instance, examining the first row, we can observe that in 48% of the Rubble images, Bulky items are also present, while in 51% of the Bulky items images, Rubble is present. Specifically, each cell represents the percentage of images in which both waste types are present relative to the total number of images where the waste identified by the respective row appears.

Finally, within the AerialWaste dataset, specifically in the testing set, 169 images are available with segmentation masks. Notably, multiple waste items may be annotated within each image, with each waste item having its waste type identified along with the

corresponding areas delineated via polygons, resulting in segmentation masks.

As can be seen from Table 3.3, only 5 out of the 15 waste types have such masks available, while the remaining types do not; hence, they have been omitted from the table. Moreover, the scarcity of masks is evident, which poses challenges for training a model to accurately locate or predict masks in a supervised manner.

	Number of masks
Rubble	128
Bulky items	69
Fire Wood	63
Vehicles	70
Tires	36

Table 3.3: The number of the instance for each type of wastes with the segmentation masks.

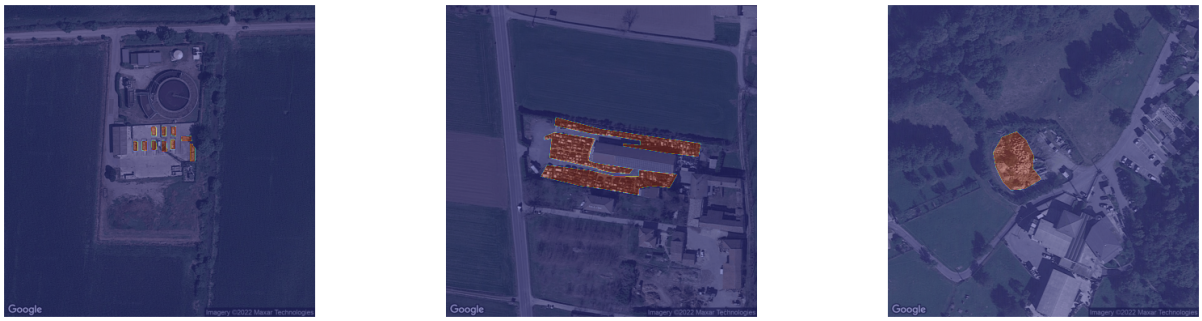


Figure 3.4: Some examples of segmentation masks.

3.2. Proposed Approaches and Methods

3.2.1. Standardizing Image Dimensions and Spatial Resolution

The dimensions and spatial resolution (cm/pixel) of the images posed an important aspect requiring careful consideration. As previously discussed, the dataset comprises images with varying dimensions and spatial resolutions. To illustrate, images from the Italian Agriculture Development Agency (AGEA) exhibit a spatial resolution of approximately 20 cm Ground Sample Distance (GSD) and dimensions of $1,000 \times 1,000$ pixels. Conversely, WorldView3 provides high-resolution images from a commercial satellite with a spatial resolution of around 30 cm GSD and dimensions of 700×700 pixels.

To train a model effectively, it's imperative that all images share the same dimensions. One might initially consider simply resizing the images to a predetermined size. However, this approach would yield images with the same dimensions but differing spatial resolutions.

Our objective was twofold: achieving both uniform dimensions and spatial resolution. To accomplish this, we leveraged our prior knowledge that each image covered a total area of 210×210 square meters. With this information, by dividing one side (21,000 centimeters) by the image size in pixels, we could compute the exact spatial resolution of each image using the following formula :

$$\frac{21000\,\text{centimeters}}{\text{image dimension in pixels}}$$

resulting in cm/pixel.

Once the spatial resolution of each image was determined, we could set a specific target spatial resolution to which we want to upsample all images. This involve resizing the images to a specific dimension, considering both the starting spatial resolution and pixel dimensions of each image, as well as the desired final spatial resolution. Specifically, the new dimension of the images targeted for resizing was calculated using the following formula:

$$\text{newShape} = \text{Shape} \times \left(\frac{\frac{21000cm}{\text{Shape}}}{\text{fixedResolution}} \right)$$

However, this would still leave us with images of varying dimensions. To address this, we applied black padding around each image to ensure uniform dimensions. Ultimately, this preprocessing step resulted in images with both uniform spatial resolution and dimensions.

Here is the summary table with the four different spatial resolutions and their respective dimensions with padding 3.4:

Spatial Resolution (cm/pixel)	Image Shape (pixel)
100 cm/pixel	224x224
50 cm/pixel	448x448
30 cm/pixel	736x736
20 cm/pixel	1088x1088

Table 3.4: Different spatial resolutions and their respective dimensions with padding.

3.2.2. Data Augmentation

As mentioned earlier, specific layers were utilized during training to execute data augmentation. Data augmentation serves as a fundamental technique in training machine learning models, aiming to enhance the diversity of training data by introducing artificial variations in images or input data. This strategy plays a pivotal role in improving the model's generalization capabilities while mitigating the risk of overfitting.

Each of the data augmentation transformations mentioned fulfills a distinct purpose in generating new data:

- RandomFlip: This transformation executes either a horizontal or vertical flip of the image with a defined probability, facilitating the model's acquisition of invariance to image mirroring.
- RandomRotation: By randomly rotating the image within a specified angle range, this transformation aids the model in developing invariance to image rotation.
- RandomNoiseLayer: Adding random noise to the image enhances the model's resilience to variability in the input data.
- RandomBrightness: This transformation randomly adjusts the brightness of the image, thereby contributing to the model's ability to handle variations in image brightness effectively.
- RandomContrast: By randomly adjusting the contrast of the image, this transformation assists the model in becoming invariant to fluctuations in image contrast.

The integration of these data augmentation transformations into the training process of a classifier with satellite images yields various benefits:

- Dataset Augmentation: Leveraging data augmentation techniques enables the generation of synthetic images from existing ones, thereby expanding the size of the training dataset. This proves particularly advantageous when the original dataset is limited.
- Generalization Improvement: The introduction of artificial variations in the images enables the model to recognize and generalize better on pertinent patterns, thus enhancing its capacity to generalize to unseen test data.
- Reduction of Overfitting Risk: By diversifying the training images, the risk of overfitting is mitigated. Consequently, the model does not merely memorize the training images but acquires the ability to discern more generalized patterns, leading to im-

proved performance on unseen data.

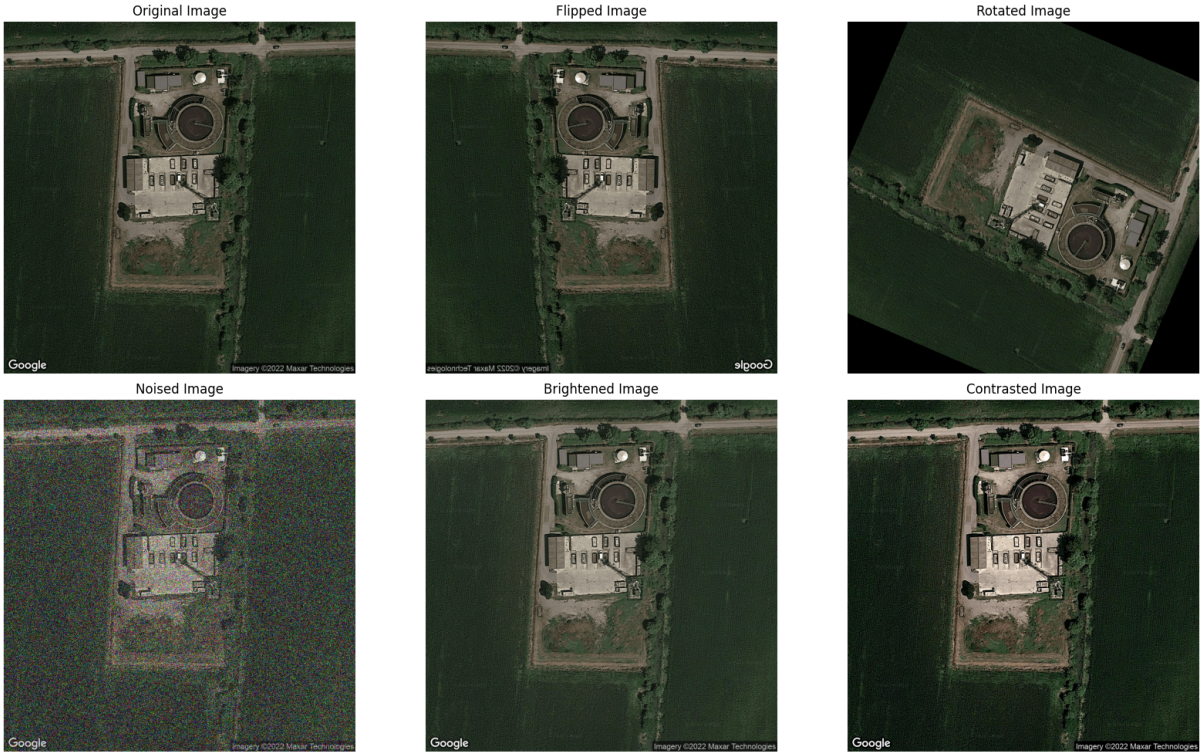


Figure 3.5: Here is a figure illustrating the results of various data augmentation transformations. From left to right and top to bottom, we have: Original image, Flipped image, Rotated image, Noisy image, Image with modified brightness, Image with modified contrast.

3.2.3. Class Activation Maps and Grad-CAM++

As addressed in previous chapters, our goal has been to localize the position of waste in a specific area. However, for this purpose, we do not have access to a dataset that would allow us to use supervised approaches, as we lack annotations such as bounding boxes or segmentation masks sufficient to pursue this path. Therefore we resorted to a weakly-supervised approach, where we train the CNN to only solve the classification task. In our case, this task involved binary classification to determine the presence or absence of waste in satellite images, and subsequently identifying the regions of the image that led the network to classify the image as positive, thus detecting the presence of waste. This approach is known as Class Activation Maps [60], which leverages trained classification models to subsequently extract areas relevant to the classification. The utility of this approach is evident, especially in a situation like ours where we do not have annotations to train supervised localization models but only have binary image-level annotations that

identify the presence or absence of waste.

Within this context, techniques such as Class Activation Maps (CAM) have garnered significant attention. CAM serves as an interpretative tool, facilitating the visualization of pertinent regions within images pivotal to the classification decisions made by the model. Moreover, advancing upon this notion, Grad-CAM++ further refines the CAM framework, catering to the complexities of modern deep convolutional neural networks [7]. By extending CAM's capabilities, Grad-CAM++ enables a heightened precision in localizing salient features within the data, thereby enhancing our understanding of model behavior and bolstering interpretability in deep learning.

Here's a detailed explanation of how Grad-CAM++ is calculated:

- Backpropagation: As the initial step, a backpropagation pass is executed through the pre-trained convolutional neural network (CNN) until the feature map of the last convolutional layer.

$$\frac{\partial y_c}{\partial A^{(k)}}$$

  where:

  - y_c is the target class score.
  - $A^{(k)}$ represents the feature maps of the last convolutional layer.

- Calculation of Importance Weights: Subsequently, importance weights are computed for each channel of the last convolutional layer's feature map. These weights reflect the relative importance of each channel in the model's output for a given class of interest. The weights are calculated as the weighted average of gradients with respect to the model's output concerning the pixels of the corresponding feature map.

$$w_i = \frac{1}{H \times W} \sum_p \sum_q \frac{\partial y_c}{\partial A_{i,j}^{(k)}}$$

  where:

  - w_i is the importance weight for channel i.
  - H and W are the height and width of the feature map.
  - $A_{i,j}^{(k)}$ represents the activation of channel i at spatial location (j, k).

- Aggregation of Importance Weights: After computing the importance weights, these weights are aggregated to obtain a single weight map for each channel of the last convolutional layer's feature map. This is done by multiplying the importance weights

with the corresponding feature map values and summing the result over all pixels of the feature map.

- Application of Weight Map on Feature Map: Finally, the aggregated weight map obtained is multiplied with the corresponding feature map of the last convolutional layer. This produces a weighted activation map, which highlights the salient and discriminative regions within the image with respect to the class of interest.

$$L^{gradCAM++} = ReLU\left(\sum_i w_i A_i^{(k)}\right)$$

where:

  - $L^{gradCAM++}$ is the Grad-CAM++ output.
  - $ReLU$ is the Rectified Linear Unit function.

- Up-sampling: an up-sampling of the weighted activation map is performed to align it with the original image dimensions. This makes it easier to overlay the activation map on the image to visualize the relevant regions.

Finally, after computing the Grad-CAM++ output, the values are normalized by dividing them by the maximum value, resulting in a scaling between 0 and 1. This normalization simulates a probability distribution, facilitating the extraction of a mask identifying the area of interest with waste through the application of a threshold. The ReLU function, which eliminates negative values by setting them to 0, makes additional normalization unnecessary. For upsampling, a simple bilinear interpolation was employed for its computational efficiency.

The architecture of the model used is as follows:

The model begins with an input layer followed by several layers that perform data augmentation. These augmentation layers are applied to the input image to enhance its diversity and improve the model's robustness to variations in the input data.

Subsequently, the augmented image is passed through a backbone, which consists primarily of convolutional layers, max-pooling layers, and activation layers. This backbone is pre-trained on the ImageNet dataset and serves as a feature extractor. Specifically, two different backbones are utilized: ResNet50 [20] and InceptionResNetV2 [52]. Each backbone has a different receptive field size, with ResNet50 having a receptive field of 483 and InceptionResNetV2 having a receptive field of 3039. The receptive field refers to the effective size of the input region that influences the output of a neuron in the

network. A larger receptive field is beneficial as it allows the network to capture more global contextual information from the input image.

Before passing the image to the backbone, preprocessing specific to each backbone is applied to the image. This preprocessing step ensures that the input data is in the appropriate format for the respective backbone architecture.

Following the backbone, the last feature map generated is extracted and passed through a Global Average Pooling layer. This layer aggregates spatial information across the feature map by taking the average of each channel's values, resulting in a one-dimensional vector. This vector is invariant to the input image's size, making it suitable for handling images of different dimensions.

Finally, four Dense layers are added to perform classification based on the features extracted by the backbone. ReLU activation functions are applied to each Dense layer, except for the last layer, which utilizes a sigmoid activation function. The sigmoid function outputs probabilities between 0 and 1, making it suitable for both binary and multi-label classification scenarios.

The pre-trained layers of the backbones are not utilized since they were trained on a different number of classes. Instead, the backbones serve as feature extractors, and the Dense layers are trained specifically for the classification task at hand.

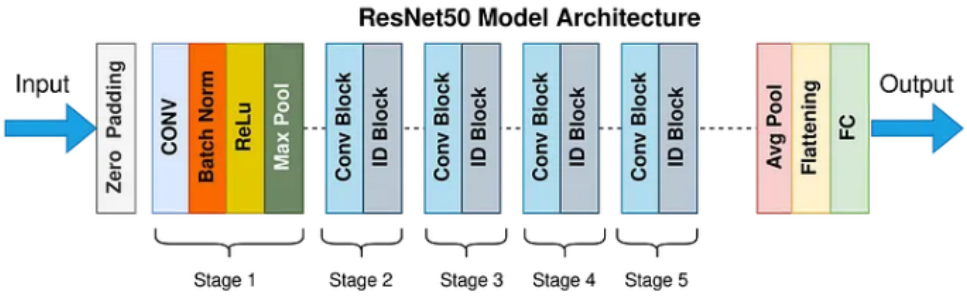


Figure 3.6: Resnet-50 Model architecture

Another important aspect to consider is the ratio of positive to negative images, as there are twice as many negative images (without waste) compared to positive images (with waste). This imbalance in the dataset may bias the model towards predicting negative outcomes more readily than positive ones. To address this issue, various methodologies can be employed, including Random Oversampling, Random Downsampling, and the use of class weights. In this case, we opted for Random Oversampling since the difference in the ratio between the two classes was minimal. We ruled out Random Downsampling a

priori because it would involve randomly discarding examples of negative images, potentially removing valuable instances that could capture important features for identifying additional backgrounds resembling waste. Below we report the algorithm used for random oversampling 3.1.

Algorithm 3.1 Random Oversampling Algorithm

1: **Input:** List of positive images P, list of negative images N
2: **Output:** Balanced dataset with equal number of positive and negative images
3: Calculate the difference between the number of negative and positive images: $diff \leftarrow |N| - |P|$
4: **if** $diff > 0$ **then**
5: Randomly select $diff$ positive images from P with replacement
6: Add the selected positive images to P until $|P| = |N|$
7: **end if**

We provide a summary table with the parameters and choices made during the training of the models 3.5:

Spatial Resolution	20/30/50/100 cm/pixel
Image Shape	1088/736/448/224 pixels
Data augmentation	Flip, Rotation, Noise, Brightness, Contrast
Backbone	ResNet50/InceptionResNetV2/ViTB16
Learning Rate	1e-5
Batch Size	10
Regularization with L2	1e-2
Balancing dataset	Random Oversampling
Version of the dataset	AerialWaste 2.0

Table 3.5: The number of the instance for each type of wastes with the segmentation masks.

3.2.4. Heatmap Generation from Feature Maps Across Different Scales

The previous approach, which involved extracting class activation maps from a classifier, revealed a well-known issue: as classifiers descend through the layers using operations like max-pooling or strides greater than 1, they need to reduce the spatial dimension of

feature maps. Consequently, the final heatmap has a different and much lower resolution compared to the original image resolution. This makes it difficult to ascertain the exact probability of each pixel in the initial resolution, often leading to approximate localization through bilinear upsampling.

To address this challenge, we proposed extracting feature maps at different scales, each with a different spatial resolution, to better identify the precise location of waste in high-resolution pixels. To achieve this, we utilized the ResNet50 backbone architecture, consisting of 5 sequential blocks. Each block performs convolution, batch normalization, ReLU, and max-pooling operations sequentially. As a result, each block generates feature maps at a different scale, with reduced spatial resolution due to the max-pooling operation. Therefore, we opted to extract feature maps from the last 4 blocks, resulting in 4 feature maps at spatial resolutions identified by symbols $s2, s3, s4$, and $s5$. Each feature map's dimensions are halved compared to the preceding ones. For instance, if our starting image has dimensions of 448x448 pixels, then $s1$ will have 224x224, $s2$ 112x112, $s3$ 56x56, $s4$ 28x28, and $s5$ 14x14 pixels.

Once we have extracted these feature maps at different scales, they will have a different number of channels because lower blocks will have more channels than the preceding ones. Our goal is to produce a heatmap, an output with a single channel with values between 0 and 1, representing the probability of waste being present at that location. To achieve this, we apply different layers of convolution with 1x1 filters to each pixel of each feature map. For the final convolution layer, we apply a sigmoid function, simulating binary classification for each pixel of each feature map. Pixels in the same feature map share the same weights, while pixels in different feature maps have different filters and therefore different weights. At the end of this process, we obtain 4 heatmaps with different resolutions.

Since our dataset lacks sufficient segmentation masks to directly train these heatmaps, the only option available is to extract from each heatmap a classification probability indicating the presence or absence of waste in the image. To do this, we could use popular Global Pooling techniques such as Global Average Pooling or Global Max Pooling. However, various studies have shown some issues with these pooling techniques: Global tends to highlight a larger area than necessary, while Max tends to highlight a smaller area. To address these issues, we employed Top-T Pooling introduced in the paper "Weakly-supervised High-resolution Segmentation of Mammography Images for Breast Cancer Diagnosis" [33], which involves averaging the first K highest probabilities in the heatmaps, where K is determined as the quantity of probabilities equal to the square of one side of the heatmap multiplied by the percentage indicating the amount of probabilities to

consider.

$$y_n = \frac{1}{|H|} \sum_{(i,j) \in H} S_n(i,j) \tag{3.1}$$

where:

- y_n the average activation value for heatmap scale n.
- $|H|$ the set containing the locations of the highest probabilities in the heatmap.
- $S_n(i,j)$ the activation value of pixel (i,j) in the heatmap of scale n.

To recap, from each of the 4 heatmaps at different scales, a prediction is extracted using Top-T Pooling. This yields 4 predictions on which we compute the error to update the model parameters. Importantly, since we force predictions from the earliest layers, the model learns relevant features for our waste identification task from the outset, not just in the later layers. This technique has been used to train numerous models, such as GoogLeNet [51].

Another crucial aspect introduced is the attempt to enforce sparsity in the heatmaps. Analyzing the distribution of positive pixels in the 169 segmentation masks available, it emerged that only an infinitesimal fraction of pixels in the high-resolution image are positive. Therefore, it's essential to compel the model to learn heatmaps that incorporate this sparsity property, in addition to correctly predicting the labels. To achieve this, along with computing the cross-entropy error for the 4 predictions as the training loss function, a term of L1 regularization on heatmap sparsity is introduced. This term consists of the simple sum of the absolute values of the probabilities of the 4 heatmaps. As is well known, L1 regularization encourages many values to be 0 to obtain sparsity.

Below, we present the formula adopted for the training loss function of the model:

$$L = \sum_{n=2}^{5} \text{BinaryCrossEntropy}(y, y_n) + \lambda \sum_{i=1}^{H} \sum_{j=1}^{W} |S_n(i,j)| \tag{3.2}$$

where:

- L the total loss function.
- n the index of the heatmap scales (from 2 to 5).
- $BinaryCrossEntropy$ the binary cross entropy between the labels y and predictions

y_n for scale n.

- λ the regularization coefficient for heatmap sparsity.
- H the height of the heatmap.
- W the width of the heatmap.
- $S_n(i, j)$ the value of pixel (i, j) in the heatmap of scale n.

Once the model has been trained, given a high-resolution image, the model provides us with four heatmaps at different scales and their corresponding predictions. To obtain the final prediction, we computed a combination of the four predictions. As for the heatmaps, the first step was to resize them to match the resolution of the original image since they had different resolutions. Then, we experimented with different aggregation methods such as multiplication or addition among them. Ultimately, we settled on a linear combination as the final solution for the heatmaps.

To identify the most suitable value for the hyperparameter concerning the number of highest probabilities to consider for averaging and determining the prediction from each heatmap, we extracted the distribution of the percentage of positive pixels in each image from the 169 available segmentation masks. As depicted in the figure 3.7, except for a few outliers, the majority of values concentrate between 0 and 5 percent. Specifically, we observed a mean of 0.0231, a median of 0.0142, a minimum value of 0.00036, and a maximum value of 0.3286. Given these properties, we focused on small values for the hyperparameter, ranging from 0.01 to 0.05.

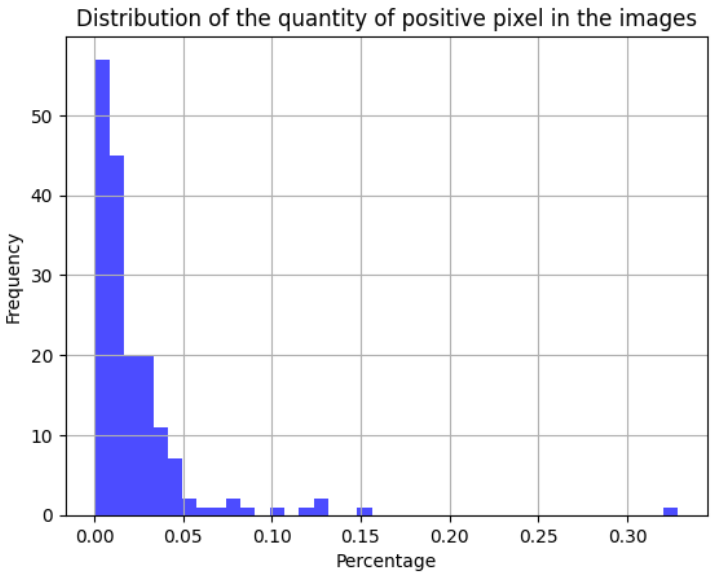


Figure 3.7: The distribution of the percentage of positive pixels in the images.

Also, for this technique, we adopted the same approaches. Specifically, we balanced the dataset using random oversampling and applied data augmentation techniques such as random flip, random rotation, random noise, random brightness, and contrast adjustments. Additionally, we ensure that all images had the same spatial resolution and pixel dimensions through black padding.

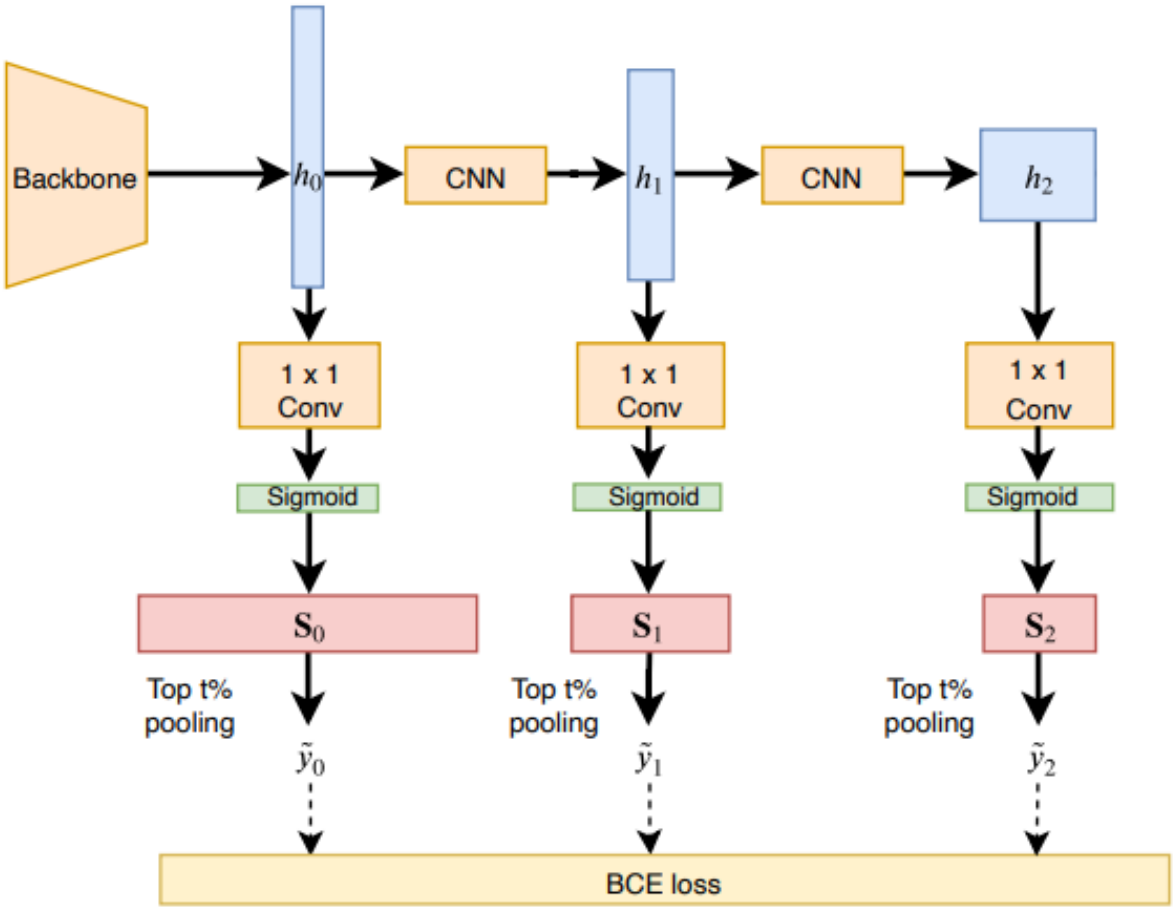


Figure 3.8: This is the architecture used in the paper from which we drew inspiration [33]. The only differences lie in the utilization of an additional block of feature maps, as we employed four instead of three compared to theirs, and the use of multiple and sequential convolutional layers with varying numbers of filters, except for the final layer which has one filter with a sigmoid function to obtain probabilities.

3.2.5. Enhancing Discriminative Regions in Heatmaps through Random Image Occlusion

Both of the previous approaches presented, namely class activation maps and heatmaps generated from feature maps at different scales, suffer from a well-known issue in the liter-

ature: they only highlight the most discriminative areas in the image. This phenomenon occurs because deep learning models, during training, tend to focus on the most relevant features for image classification. Therefore, when generating class activation maps or heatmaps, the models primarily highlight the image regions that contribute significantly to the classification decision, neglecting background and context information that is less relevant to the classification task. In other words, these approaches emphasize the regions of the image that have a greater impact on the model's prediction but do not provide a complete representation of the entire image.

To address this issue, we drew inspiration from the work presented in the paper titled "Hide-and-Seek: Forcing a Network to be Meticulous for Weakly-supervised Object and Action Localization" by Krishna Kumar Singh and Yong Jae Lee [50]. In their work, the authors propose a strategy where certain patches of the image are randomly hidden during training. By doing so, the model is exposed to different relevant parts of the image at different times, forcing it to thoroughly examine various regions for accurate classification. Consequently, the model learns to explore additional relevant areas when discriminative regions are hidden. It's important to note that during inference, there's no need to hide any areas, as the model is expected to identify all relevant regions for classification independently .

In particular, the strategy we implemented is as follows: given an image of size WxHx3, we first divide it into a grid with a fixed patch size of S×S×3, where S represents the size of each patch. This grid division results in a total of (W × H)/(S × S) patches. For each patch, we generate a probability value between [0,1] from a uniform distribution. Subsequently, a patch is obscured if its probability exceeds a certain threshold that we set. Therefore, the essential hyperparameters are two: the patch size and the probability threshold for obscuring.

Regarding the patch size, we opted for a small value of 16x16 with a resolution of 50 cm/pixel and 32x32 with a resolution of 20 cm/pixel. This choice was made because, as analyzed earlier, positive pixels (representing waste areas) tend to be very small. Hence, using large patches might lead to completely obscuring the waste area, resulting in false positives during training.

As for the probability threshold, we tested values of 0.9, 0.8, and 0.7. We focused on these values because smaller probabilities might generate false positives.

Below is the algorithm employed to randomly obscure patches during the training phase 3.2:

Algorithm 3.2 Patch Obscuring Algorithm

Input: Image I, Patch size S, Probability threshold P
Output: Image with obscured patches I'
Divide I into a grid of patches of size $S \times S \times 3$
for each patch p **do**
    Generate a random probability *prob* between [0,1]
    if $prob > P$ **then**
        Obscure patch p in image I
    end if
end for

Below is a visual example of the effect on the original image due to obscuring 16x16 patches with different threshold probabilities applied for obscuration:

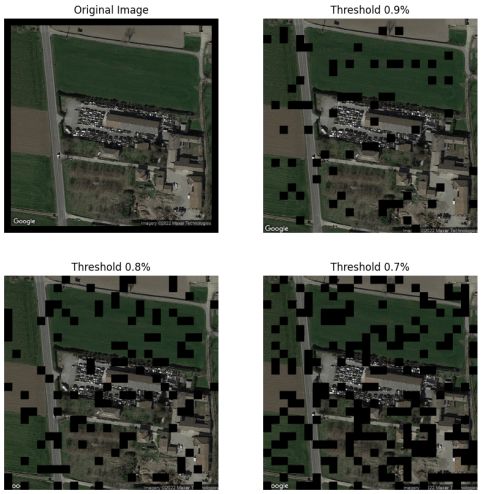


Figure 3.9: Example of Patch Obscuring with different threshold.

3.2.6. Multi Instance Learning

Unlike the previous three approaches we have discussed, which are cam-based, meaning they rely on creating activation maps highlighting important regions of the image for classification, with this latest proposed approach, we have delved into an alternative learning paradigm called mil-based.

Multi Instance Learning (MIL) is a different learning paradigm where a data instance, such as an image, is represented as a set of sub-instances rather than as a single entity. Of course, the class label of an instance is known, but the labels of the sub-instances within that instance are not. In our specific case, considering the binary scenario, the

HxWx3 image is the starting instance associated with a binary label, positive or negative, indicating the presence of waste or not. Meanwhile, patches extracted from it would represent sub-instances, and their collection would constitute a bag still associated with the original binary label.

Given an HxWx3 image and a specified patch size PxPx3, different approaches can be used to extract the patches. One possible approach could be to divide the starting image into a grid where each cell corresponds to a PxPx3 patch. Therefore, the total number of non-overlapping patches generated would be H/P * W/P = N. These N patches compose the representative bag of the image, along with its associated starting label, for model training. Naturally, various approaches exist in the literature for generating patches from the starting image. Some differences may include whether patches overlap, the number of patches used to compose the bag (all extracted patches or a certain number K), and different methods for selecting the K patches. For example, one may utilize heatmaps produced by a previous model, indicating areas with a higher likelihood of waste presence or areas of greater interest/difficulty.

The extraction of patches/sub-instances to form the bag from the starting image/instance along with its corresponding label constitutes the first phase of the MIL approach. The second phase involves individually processing each patch using a computer vision model, such as a CNN like ResNet50, which converts the patches into patch-level feature representations using a one-dimensional vector for each. In this case, it suffices to take the last feature maps of the CNN and apply Global Pooling to make them one-dimensional, for example, using Global Average Pooling.

After this second phase, we will have a 1D vector for each patch extracted from the starting image, and our objective remains to classify the entire starting image since we have the associated label for it and not for the individual extracted patches. Therefore, as the third phase, it is necessary before proceeding with the classification to aggregate the set of patch-level feature representations into a single feature representation for the entire image. This can be achieved using a specific pooling technique. As highlighted in the section 2.2.4 regarding MIL, there can be different types of pooling in this phase, which can be divided into static ones like Average or Max Pooling, or adaptive ones where through learning techniques, we determine the relevance of each patch for classification, quantifying how relevant and useful its information/representation is. Regardless of which pooling technique is used, the final result is the same, i.e., having a single 1D representation derived from the K 1D representations of the patches.

Finally, as the last phase, this single representative 1D representation of the initial image

or bag of extracted patches is fed into a classification head primarily composed of Dense Layers. The output consists of the final score, indicating the probability of waste presence or absence in our case.

In this thesis, we have implemented the second phase in the following way:

In the first scenario, given the initial image of size HxWx3 and a specified patch size of PxPx3, we divided the starting image into a grid where each cell corresponds to a PxPx3 patch. Therefore, the total number of non-overlapping patches generated would be H/P * W/P = N, and the corresponding bag for an image would be the set of all possible non-overlapping extracted patches.

Going into more detail, we considered two spatial resolutions for the starting images, namely 20 and 50 cm/pixel, with their respective dimensions in pixels with black padding being 1024x1024 and 512x512. Subsequently, for each image with different resolutions, we extracted patches using the aforementioned method with dimensions in pixels of 32x32, 64x64, 128x128, 256x256, and 512x512 (the latter only for the 20 cm/pixel resolution, as for the 50 cm/pixel resolution it was not possible to extract more patches since the maximum starting size was 512x512).

Below is a summary table 3.6 of the spatial resolutions and dimensions adopted, along with the corresponding number of non-overlapping patches extracted for a given size:

Spatial Resolution (cm/pixel)	Image Shape (pixel)	Patch Size (pixel)	Number of patches
20 cm/pixel	1024x1024	32x32	1024
20 cm/pixel	1024x1024	64x64	256
20 cm/pixel	1024x1024	128x128	64
20 cm/pixel	1024x1024	256x256	16
20 cm/pixel	1024x1024	512x512	4
50 cm/pixel	512x512	32x32	256
50 cm/pixel	512x512	64x64	64
50 cm/pixel	512x512	128x128	16
50 cm/pixel	512x512	256x256	4

Table 3.6: Spatial resolutions and dimensions adopted, along with the corresponding number of non-overlapping patches extracted for a given size.

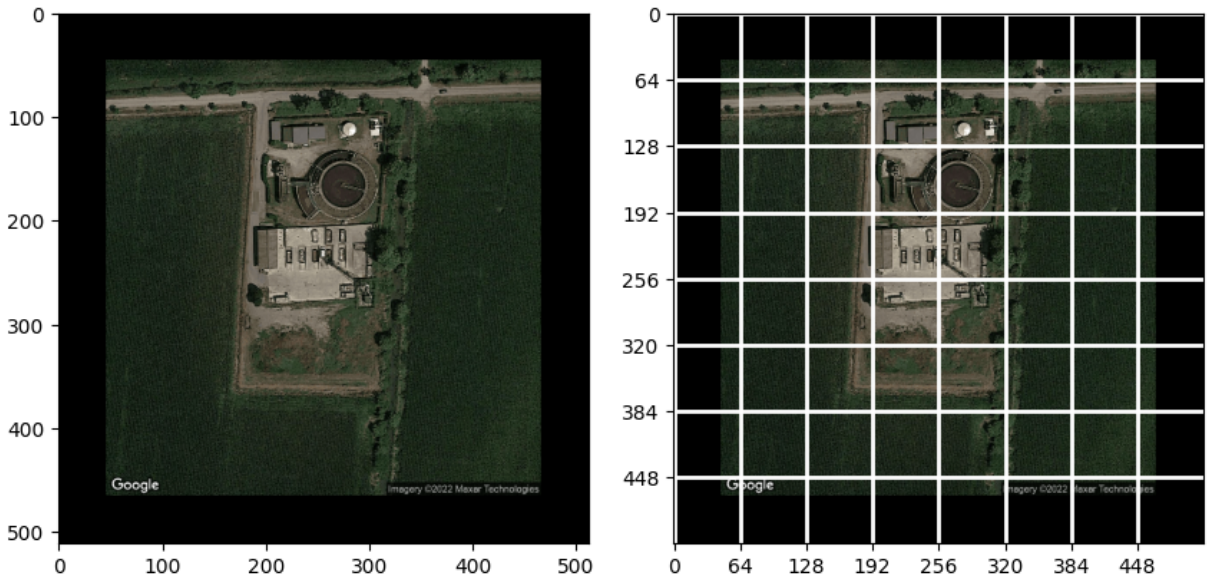


Figure 3.10: Example of patches 64x64 from an image with 50 cm/pixel and 512x512 pixels.

In the second scenario, we aimed to utilize only a subset of the extracted images from the set of all possible non-overlapping patches extracted of a certain size. In this scenario, the real challenge lies in determining which patches to extract, as we do not have access to the labels of each patch or a segmentation mask to identify areas of interest. To address this, we relied on the heatmaps extracted using the approach introduced in section 3.2.2 "Heatmap Generation from Feature Maps Across Different Scales" where we generated a final heatmap by linearly combining 4 heatmaps generated from feature maps extracted from different layers and scales.

Therefore, given an image, for example, 512x512 with 50 cm/pixel resolution, it is provided to the model to generate its predicted heatmap, which highlights areas with a higher likelihood of containing waste. Then, for each patch extracted from the original image, its corresponding patch from the predicted heatmap is also extracted, and its score is calculated by averaging all the probability values present in that area, or in that patch. Subsequently, the patches are sorted along with their corresponding probability scores in descending order, and finally, the top K patches with the highest probabilities of containing waste are used to compose the bag.

Below is the algorithm used to extract a subset K of possible patches and a corresponding visual example of its operation:

Algorithm 3.3 Patch Selection Algorithm using Heatmap

Input: Starting image I with dimensions $H \times W \times 3$, patch size $P \times P \times 3$
Output: Bag of selected patches
Generate heatmap H for image I using CNN
for each patch p extracted from I **do**
    Extract corresponding patch h from H
    Calculate score s_p for patch p based on the average of probabilities in patch h
end for
Sort patches p based on scores s_p in descending order
Select top K patches with highest scores to compose the bag

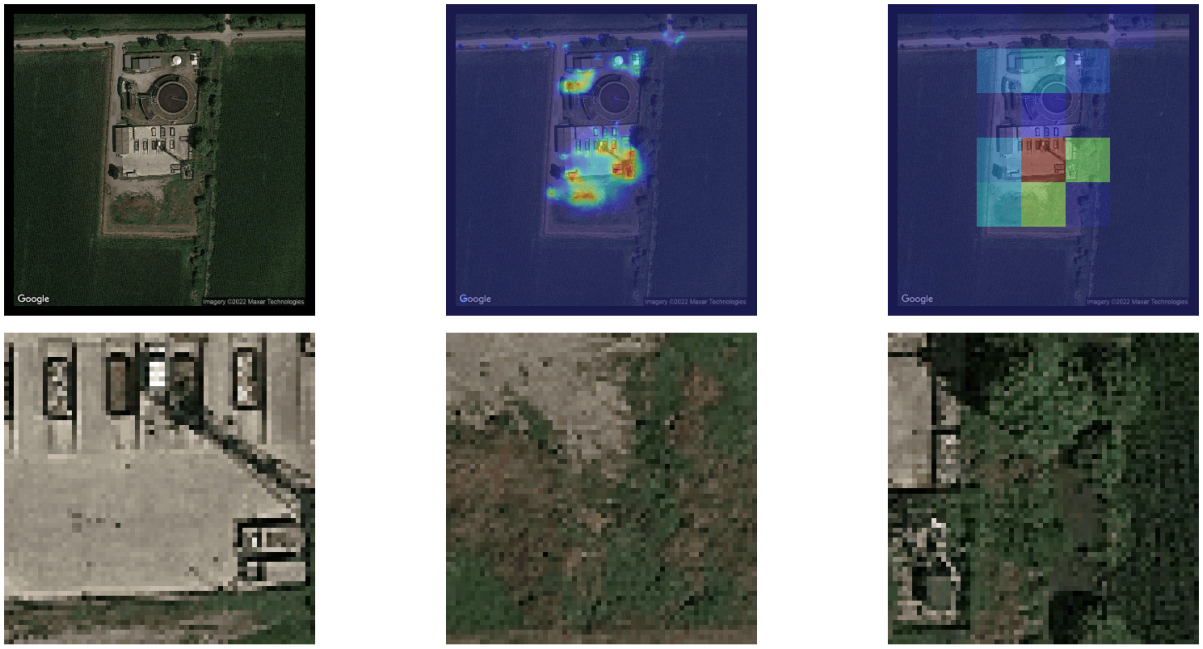


Figure 3.11: Example of an image with its predicted heatmap and the relative scores for each patch in the first row while in the second row we can see the first 3 patches extracted with the highest scores.

Afterwards, given the bag of instances composed of each selected patch, each individual patch is processed using a ResNet50. Specifically, the last feature map produced by the backbone is extracted, and Global Average Pooling is applied, calculating the mean value for each channel. In this manner, we obtain a single one-dimensional representation for each patch, represented by a 1D vector.

Next, given the K 1D vectors, one for each patch, where K is the number of patches, they are aggregated using the concept of a gated attention mechanism introduced in the

paper 'Attention-based Deep Multiple Instance Learning' by Maximilian Ilse, Jakub M. Tomczak and Max Welling [24]. This mechanism enables pooling and aggregation among these vectors by learning and assigning importance to each patch. In other words, to aggregate the vectors into a single vector, a weighted average is performed, and the weights represent the importance of each patch calculated through this mechanism. Specifically, they proposed to use a weighted average of instances (low-dimensional embeddings) where weights are determined by a neural network and, additionally, the weights must sum to 1 to be invariant to the size of a bag.

$$z = \sum_{k=1}^{K} a_k \cdot h_k \tag{3.3}$$

where:

- z represents the resulting aggregated vector.
- $\sum_{k=1}^{K}$ denotes the summation over all values of k from 1 to K, where K is the number of patches.
- a_k are the weights associated with each patch k.
- h_k is the embedding (representation) of patch k.

While the weights for each patch are calculated as follows:

$$a_k = \text{softmax}(w^\top \tanh(V h_k^\top) \odot \sigma(U h_k^\top)) \tag{3.4}$$

where:

- $\odot$ element-wise product.
- a_k are the weights associated with each patch k
- h_k is the embedding (representation) of patch k.
- w, U and V are parameters

Finally, after pooling the K 1D vectors of the K patches, we obtain a single representative 1D vector of the patch bag. This vector is then provided to the classification head, primarily composed of several Dense Layers, which makes predictions to determine the probability of waste presence or absence in the image.

Also, for this technique, we adopted the same pre-processing as in the previous sections. Specifically, we balanced the dataset using random oversampling and applied data

augmentation techniques such as random flip, random rotation, random noise, random brightness, and contrast adjustments. Additionally, we employed preprocessing steps to ensure that all images had the same spatial resolution and pixel dimensions through black padding.

After training the model, it is possible to perform inference with any number of patches composing the bag since, as mentioned earlier, the softmax makes the process independent of the bag size. In particular, it is possible to classify each individual patch simply by constructing bags of single patches. Therefore, given a patch, it will pass through the ResNet50, generating a 1D vector representing it. Subsequently, its weight is calculated and normalized. However, since it is only one patch, its normalization through softmax ensures that the normalized weight is 1, and thus the 1D vector remains unchanged and can be provided to the classification head, which will provide the probability of that patch.

Therefore, given a patch, it is possible to obtain its weight before normalization and its probability. With these two scores, it is possible to create a map of relevant areas in the original image simply by extracting every possible patch, possibly using a reduced shift for more accurate localizations. In our case, we use a shift equal to the patch size divided by 8. For each extracted patch, we calculate the importance weight and the probability of containing waste. Subsequently, we assign to the pixels falling within a certain patch its probability or its importance weight, or a fusion of the two. Logically, in the case of overlapping patches, we will sum the values determined for a specific pixel present in different patches, and then divide this sum by the number of patches in which it was present to obtain an average score of all the patches where it was present.

Algorithm 3.4 Calculation of sp, wp, and swp

Input: Starting image I with dimensions $H \times W \times 3$, patch size $P \times P \times 3$, local_probability are the probabilities of the patches, weights are the attention weights of the patches
Output: sp, wp, and swp: arrays for sum of probabilities, weights and fusion between probabilities and weights
for each patch extracted from the image **do**
    $sp[i : i + \text{patch\_size}, j : j + \text{patch\_size}, :] += \text{local\_probability}[\text{count\_patch}, 0]$
    $wp[i : i + \text{patch\_size}, j : j + \text{patch\_size}, :] += \text{weights}[\text{count\_patch}, 0]$
    $w[i : i + \text{patch\_size}, j : j + \text{patch\_size}, :] += 1$
end for
$sp /= w$
$wp /= w$
$wp = ReLU(wp)/max(wp)$
$swp = sp * wp$

4 | Experiments and performance evaluation

4.1. Evaluation metrics

In this study, we assessed the performance of waste localization in conjunction with binary classification, indicating the presence or absence of waste in images. For classification, we utilized Precision, Recall, F1-Score, and Accuracy metrics, with particular attention given to the F1 Score. For localization, we employed IoU (Intersection over Union), Dice, Precision, Recall, and F1-Score metrics.

Binary Classification Metrics:

- Precision measures the fraction of positively classified instances that are actually positive.

$$\text{Precision} = \frac{TP}{TP + FP}$$

  Where:

  - TP (True Positives): the number of positive instances correctly classified as positive.
  - FP (False Positives): the number of negative instances incorrectly classified as positive.

- Recall measures the fraction of positive instances that were correctly detected.

$$\text{Recall} = \frac{TP}{TP + FN}$$

  Where:

  - TP (True Positives): the number of positive instances correctly classified as positive.
  - FN (False Negatives): the number of positive instances incorrectly classified as

negative.

- F1-score is the harmonic mean of precision and recall and provides a balance between the two metrics.

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- Accuracy measures the overall fraction of correct predictions.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

  Where:

  - TP (True Positives): the number of positive instances correctly classified as positive.
  - FP (False Positives): the number of negative instances incorrectly classified as positive.
  - FN (False Negatives): the number of positive instances incorrectly classified as negative.
  - TN (True Negatives): the number of negative instances correctly classified as negative.

Object Localization Metrics:

- IoU measures the overlap between the predicted area and the ground truth area of the object.

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

- Dice coefficient is a measure of similarity between two samples and is often used for image segmentation evaluation.

$$\text{Dice} = \frac{2 \times |X \cap Y|}{|X| + |Y|}$$

  Where:

  - X and Y represent the predicted area and the ground truth area of the object respectively.

- Precision, Recall, and F1-Score for Localization: These metrics are defined as above but applied to the object localization context.

4.2. Results

Initially, we assessed the proposed preprocessing to standardize the images to a single dimension and spatial resolution by simply training a CNN followed by Global Average Pooling and some Dense Layers, with the last Dense layer utilizing the sigmoid function to classify the image. Alternatively, we employed the Vision Transformer Base with 16x16 pixel patches. As for the CNN, we tested two backbones (ResNet50 and Inception-ResNetV2) to highlight any existing limitations or potentials between them. We tested images with resolutions of 20 cm/pixel, 30 cm/pixel, and 50 cm/pixel (for ViTB16, we had to stop at 30 cm/pixel due to computational reasons).

Below are the performance results achieved with ResNet50 as the backbone using different spatial resolutions and input image sizes:

Shape	Spatial Resolution	Accuracy	Precision	Recall	F1-Score
224x224	100 cm/pixel	85.57%	73.38%	88.80%	80.36%
448x448	50 cm/pixel	88.41%	82.19%	83.14%	82.66%
736x736	30 cm/pixel	88.25%	81.89%	83.03%	82.45%
1088x1088	20 cm/pixel	86.56%	78.48%	82.10%	80.25%

Table 4.1: The performance results achieved with ResNet50.

While, these are the performance results achieved with InceptionResNetV2 as the backbone using different spatial resolutions and input image sizes:

Shape	Spatial Resolution	Accuracy	Precision	Recall	F1-Score
448x448	50 cm/pixel	88.29%	84.00%	80.02%	81.96%
736x736	30 cm/pixel	90.25%	82.35%	89.95%	85.98%
1088x1088	20 cm/pixel	90.36%	82.27%	90.53%	86.20%

Table 4.2: The performance results achieved with InceptionResNetV2.

Lastly, the performance achieved with the Vision Transformer with a 16x16 patch size using different spatial resolutions and input image sizes:

Shape	Spatial Resolution	Accuracy	Precision	Recall	F1-Score
224x224	100 cm/pixel	88.64%	78.44%	90.76%	84.15%
448x448	50 cm/pixel	90.79%	82.40%	91.92%	86.90%
736x736	30 cm/pixel	92.44%	85.70%	92.73%	89.07%

Table 4.3: The performance results achieved with ViTB16.

As observed from the results (Tables 4.1, 4.2, 4.3), while InceptionResNetV2 and ViTB16 achieved the best performance with the maximum computationally usable resolution, i.e., 20 cm/pixel and 30 cm/pixel respectively, ResNet50 was limited to 50 cm/pixel. Therefore, we can infer a limitation on the part of ResNet50 as a backbone in fully utilizing the 20 cm/pixel resolution. The reason could be attributed to the fact that the receptive field of ResNet50 is 483x483, hence with an image at 20 cm/pixel with dimensions of 1088x1088, it fails to capture a global context.

Despite the limitation identified for ResNet50, we continued to focus on this backbone both to maintain a comparison with previous works and for computational reasons.

To extract waste localization, we initially utilized class activation maps through GRAD-CAM++. This method highlights the regions of an image that are influential in the decision made by a convolutional neural network. GRAD-CAM++ works by computing the gradient of the target class score with respect to the final convolutional feature map. This gradient is then global-average-pooled to obtain the importance weights for each feature map channel, which are subsequently used to weigh the feature maps, generating the final heatmap.

Backbone	Shape	Spatial Resolution	IoU	Dice	Precision	Recall	F1-Score
ResNet50	224x224	100 cm/pixel	11.29%	18.77%	15.22%	43.33%	22.53%
ResNet50	448x448	50 cm/pixel	13.87%	22.43%	17.95%	48.25%	26.17%
ResNet50	736x736	30 cm/pixel	14.33%	23.15%	22.92%	36.66%	28.20%
ResNet50	1088x1088	20 cm/pixel	18.87%	29.57%	38.49%	32.50%	35.25%

Table 4.4: The localization performance achieved with ResNet50 and GradCAM++ results.

As observed from the results presented in Table 4.4, several interesting aspects can be noted. Despite the limitation of ResNet50 in binary classification, which does not allow for better results beyond 50 cm/pixel, in the case of waste localization, we observe that the Intersection over Union (IoU) is not limited to 50 cm/pixel but continues to improve down to 20 cm/pixel. This is attributed to the fact that increasing the resolution significantly enhances precision, owing to the higher quality of details and objects, enabling more accurate identification and localization of object contours while avoiding highlighting the surrounding background. Another interesting aspect is the increase and subsequent decrease in recall, likely due to the common issue with heatmaps where models tend to highlight mainly the most discriminative areas. This is more pronounced with higher spatial resolution, altering the initial shape of the image and putting the model under

greater difficulty in identifying all areas of interest due to its reduced receptive field. Finally, the Dice coefficient and F1-Score, as expected, follow the observed behavior of the IoU.

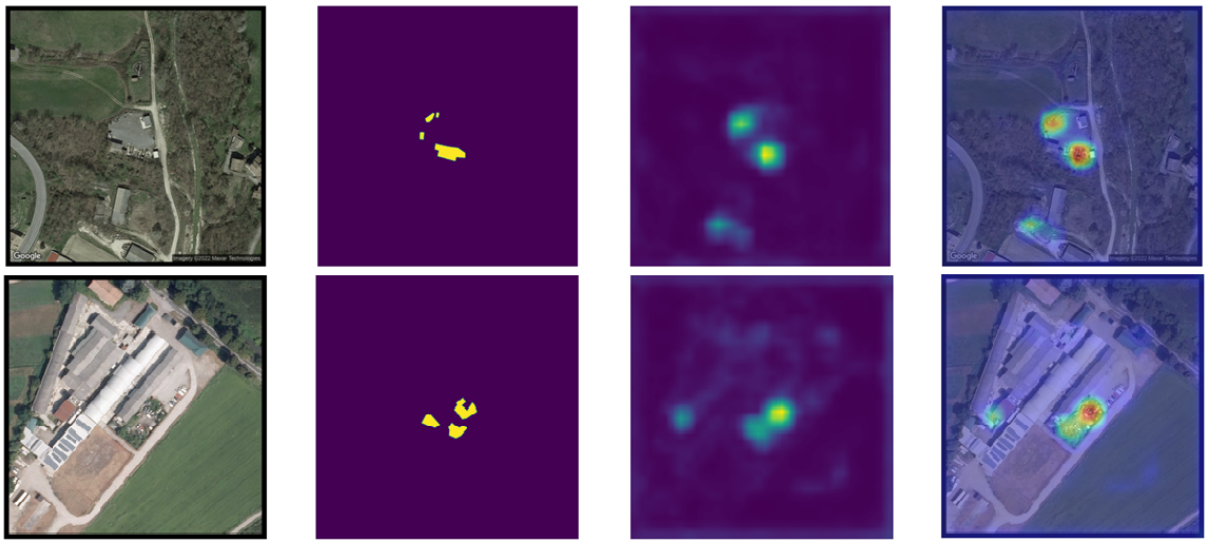


Figure 4.1: Some examples of localization obtained using GRAD-CAM++ with ResNet50. The first is the image provided as input to the model, the second is the segmentation mask annotated by a human, the third is the class activation map extracted using GRAD-CAM++ with values between [0,1], and finally we have the overlay of the image with the class activation map to identify the relevant areas with waste.

As we can observe from the above images depicting waste localization using GRAD-CAM++, it is evident that this technique allows for effective waste localization. However, we completely lose information regarding the contours and shapes of the waste. This is due to the fact that GRAD-CAM++ utilizes the last feature maps of the CNN, which have a much lower resolution compared to the initial image. This leads to the loss of information on waste boundaries as the information is compacted due to MaxPooling layers or convolutions with strides greater than 1.

To address this issue and improve both classification and waste localization performance, we have extracted not only the last feature maps but also feature maps from earlier layers, which have different spatial resolutions and dimensions. At each operation block, the feature maps are halved in size. Convolutional operations of 1x1 are applied to these feature maps at different resolutions, with a sigmoid function applied last to obtain the heatmap. Finally, the heatmaps are combined to generate the final heatmap.

This approach allowed us to achieve improvements in both tasks. By extracting feature maps from early layers, we can better identify object boundaries, while using the last fea-

ture maps helps localize the area of interest relative to the task. Additionally, predictions are extracted from each of the 4 heatmaps using TopT Pooling, which averages the K highest probabilities. These probabilities are then used in the calculation of the training loss function to update the model parameters during training. By forcing predictions from the earliest layers, the model learns relevant features for our waste identification task from the outset, not just in the later layers, leading to better classification performance.

Classification results with ResNet50 are obtained by performing inference with each prediction made from each of the 4 heatmaps generated by the 4 feature maps extracted from different layers:

Layer	Shape	Spatial Resolution	Accuracy	Precision	Recall	F1-Score
Layer 2	448x448	50 cm/pixel	75.32%	59.39%	81.41%	68.68%
Layer 3	448x448	50 cm/pixel	82.99%	69.49%	87.07%	77.29%
Layer 4	448x448	50 cm/pixel	87.37%	76.04%	90.53%	82.66%
Layer 5	448x448	50 cm/pixel	89.64%	81.97%	88.22%	84.98%
Layer 2	1088x1088	20 cm/pixel	75.36%	58.55%	88.57%	70.50%
Layer 3	1088x1088	20 cm/pixel	81.57%	67.08%	87.53%	75.95%
Layer 4	1088x1088	20 cm/pixel	86.14%	74.78%	87.99%	80.85%
Layer 5	1088x1088	20 cm/pixel	87.45%	78.76%	85.22%	81.86%

Table 4.5: Classification results obtained with ResNet50 using images with spatial resolutions of 50 and 20 cm/pixel from different layers.

Localization results with ResNet50 are obtained by combining the 4 heatmaps generated from the 4 feature maps extracted from different layers:

Shape	Spatial Resolution	IoU	Dice	Precision	Recall	F1-Score
448x448	50 cm/pixel	21.72%	33.04%	34.02%	45.94%	39.09%
1088x1088	20 cm/pixel	24.15%	36.52%	41.46%	44.25%	42.81%

Table 4.6: Localization results obtained with ResNet50 using images with spatial resolutions of 50 and 20 cm/pixel.

As observed from the results presented in the tables above, significant improvements are achieved in both classification and segmentation performance. This is attributed to the utilization of feature maps from early layers for segmentation, which helps in preserving information about waste contours and shapes, while the use of later layers ensures their localization and area determination. Regarding classification, enhancements are attained by enforcing the model to make accurate predictions from early layers, compelling it to

learn and extract relevant features right from the start, rather than solely relying on later layers.

Furthermore, in classification, we notice the possibility of performing inference and classification using predictions extracted from each layer, both from the earliest and the latest ones. However, it's evident that as we descend through the layers, precision increases, while recall increases when using a spatial resolution of 50 cm/pixel and decreases with a resolution of 20 cm/pixel, likely due to the model's struggle with the larger image size. Moreover, F1-Score and accuracy exhibit similar behavior to precision, increasing as we move down the layers. Hence, it's concluded that for classification, utilizing the last layer is preferable as it achieves higher precision, consequently leading to higher F1-Score. Additionally, the best performance is consistently achieved with a spatial resolution of 50 cm/pixel, owing to the limitations imposed by ResNet50, particularly its reduced receptive field to capture a global context present in images with 20 cm/pixel.

Concerning the localization task, similar to the previous technique, it's evident that better localization is attained using a 20 cm/pixel resolution compared to 50 cm/pixel. This is because higher resolution enables the acquisition of higher-quality images, thereby extracting more precise information regarding waste contours and shapes, resulting in a significant improvement in precision and consequently IoU, Dice, and F1 Score.

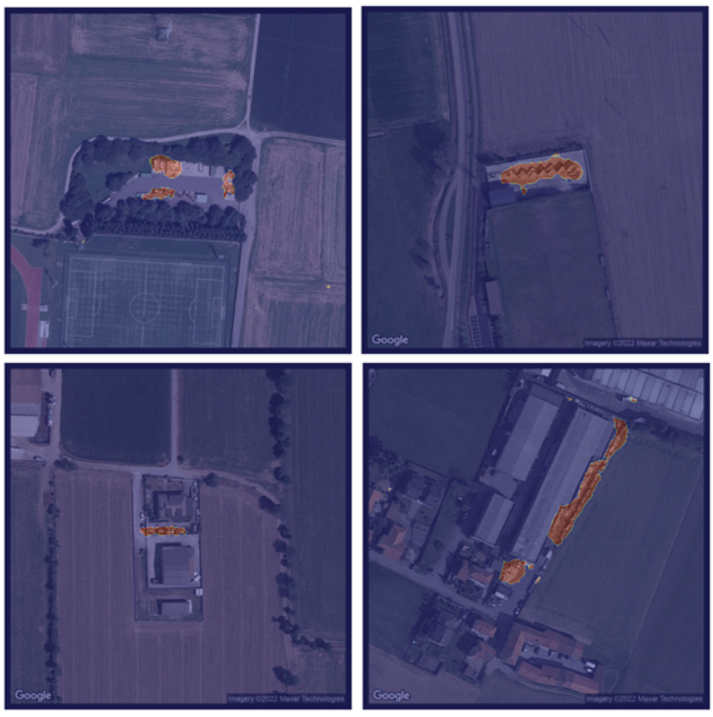


Figure 4.2: Some segmentation masks generated by combining a final heatmap obtained by combining the heatmaps generated from extracting feature maps from different layers.

Subsequently, we applied the Patch Obscuring approach, which involves obscuring a certain number of very small patches in the original image. This is done with the aim of addressing the general issue of heatmaps, which tend to focus on the most relevant features. When generating class activation maps or heatmaps, models primarily highlight image regions that significantly contribute to the classification decision, neglecting background and context information that may be less relevant to the classification task. In other words, these approaches emphasize regions of the image that have a greater impact on the model's prediction but do not provide a complete representation of the entire image.

By hiding random patches during training, the model is exposed to different relevant parts of the image at different times, compelling it to thoroughly examine various regions for accurate classification. Consequently, the model learns to explore additional relevant areas when discriminative regions are hidden. This helps in achieving a more comprehensive understanding of the image and improves the model's ability to make accurate predictions by considering a broader context.

Classification results with ResNet50 are obtained by performing inference with each prediction made from each of the 4 heatmaps generated by the 4 feature maps extracted from different layers. In the following table, we report the classification performance using predictions from the last layer since, as previously demonstrated, they yield the best results:

Shape	Spatial Resolution	Accuracy	Precision	Recall	F1-Score
448x448	50 cm/pixel	90.33%	85.05%	86.03%	85.53%
1088x1088	20 cm/pixel	87.06%	76.01%	89.26%	82.10%

Table 4.7: Classification results obtained with ResNet50 using images with spatial resolutions of 50 and 20 cm/pixel from the last layer.

Localization results with ResNet50 are obtained by combining the 4 heatmaps generated from the 4 feature maps extracted from different layers:

Shape	Spatial Resolution	IoU	Dice	Precision	Recall	F1-Score
448x448	50 cm/pixel	21.94%	33.59%	33.55%	46.14%	38.85%
1088x1088	20 cm/pixel	24.22%	36.88%	39.37%	46.84%	42.78%

Table 4.8: Localization results obtained with ResNet50 using images with spatial resolutions of 50 and 20 cm/pixel.

Observing the results of the data augmentation technique, namely patch obscuring, we can see that it enables us to achieve slight improvements both in terms of classification and localization.

Finally, we tested the MIL-based approach, where instead of predicting the presence of waste in a single image, we label sets of extracted patches. As explained in the previous chapters, we employed two approaches to compose the bags and extract patches from the image. The first approach extracts all possible non-overlapping patches of a given size PxP from the original image using a grid, while the second approach selects only a subset of the possible patches extracted with the first approach. This selection is based on using patches with the highest probability of containing waste, determined using the heatmaps generated by previous improved models.

Once the MIL-based model is trained, it can be used to classify an image by extracting the same number and in the same manner the patches from that image as done during training. Alternatively, the model can classify a set of any cardinality of patches or even classify a single patch, as the model is independent of patch size. Additionally, for each patch, it is possible to extract the importance weight assigned to it by the gate attention mechanism. This weight or the associated probability for each patch can be used to generate a heatmap simply by extracting overlapping patches with a reduced shift. For every pixel falling within such a patch, its weight or probability is assigned. As overlapping patches can enhance heatmap accuracy and object contouring, it's necessary to average the weights or probabilities of a pixel that falls into different patches.

Classification results with ResNet50 are obtained by extracting and using all possible non-overlapping patches from the image to compose the bag of patches:

Image Shape	Spatial Resolution	Patch Size	K Patches	Accuracy	Precision	Recall	F1-Score
512x512	50 cm/pixel	32x32	256	81.69%	67.93%	85.10%	75.55%
512x512	50 cm/pixel	64x64	64	82.57%	68.01%	89.84%	77.41%
512x512	50 cm/pixel	128x128	16	87.68%	77.55%	88.57%	82.70%
512x512	50 cm/pixel	256x256	4	88.75%	79.38%	89.38%	84.08%
1024x1024	20 cm/pixel	32x32	1024	81.54%	74.97%	66.74%	70.62%
1024x1024	20 cm/pixel	64x64	256	81.65%	73.10%	70.90%	71.98%
1024x1024	20 cm/pixel	128x128	64	82.30%	73.52%	73.09%	73.31%
1024x1024	20 cm/pixel	256x256	16	85.11%	77.60%	77.60%	77.60%
1024x1024	20 cm/pixel	512x512	4	87.60%	80.82%	82.22%	81.51%

Table 4.9: Classification results obtained with ResNet50 using patches with spatial resolutions of 50 and 20 cm/pixel.

Localization results with ResNet50 are obtained by extracting and using all possible non-overlapping patches from the image to compose the bag of patches and generating a heatmap by classifying each patch individually:

Image Shape	Spatial Resolution	Patch Size	Used Shift	IoU	Dice	Precision	Recall	F1-Score
512x512	50 cm/pixel	32x32	8	16.97%	26.69%	27.09%	43.75%	33.46%
512x512	50 cm/pixel	64x64	8	17.95%	28.33%	27.66%	42.86%	33.62%
512x512	50 cm/pixel	128x128	8	10.48%	17.40%	14.43%	42.36%	21.53%
512x512	50 cm/pixel	256x256	8	1.84%	3.46%	1.89%	88.39%	3.71%
1024x1024	20 cm/pixel	32x32	8	25.34%	37.92%	39.30%	49.24%	43.72%
1024x1024	20 cm/pixel	64x64	8	22.74%	34.53%	34.39%	48.53%	40.25%
1024x1024	20 cm/pixel	128x128	8	19.59%	29.89%	33.08%	37.32%	35.07%
1024x1024	20 cm/pixel	256x256	8	12.23%	20.02%	16.38%	55.29%	25.28%
1024x1024	20 cm/pixel	512x512	8	2.75%	5.09%	2.77%	88.93%	5.37%

Table 4.10: Localization results obtained with ResNet50 using patches with spatial resolutions of 50 and 20 cm/pixel.

From the results reported in the above tables, we observe that in terms of classification, we continue to achieve better results with a spatial resolution of 50 cm/pixel compared to 20 cm/pixel. Additionally, we notice that increasing the patch size, and consequently reducing the number of patches composing the bag of patches, leads to improved classification performance. However, these results do not allow us to determine whether both of these characteristics impact performance or if only one of them does.

On the other hand, examining the localization results, we can conclude that the best results are obtained with both 20 and 50 cm/pixel resolutions using small patch sizes, such as 32x32 or 64x64. This outcome is logical because significantly smaller patches enable us to more accurately identify relevant areas containing waste, thereby determining more precise contours. Indeed, precision shows a decreasing trend with increasing patch sizes, while recall exhibits the opposite behavior, increasing with larger patch sizes as they cover more area.

Another observation is that despite achieving the best classification results with 50 cm/pixel resolution, the best localization results are obtained with 20 cm/pixel resolution. This is due to the fact that with such resolution, we have higher-quality images, allowing for better definition of waste boundaries and thus obtaining greater precision in localization.

Lastly, we tested the MIL-based approach by selecting a subset of patches from the set of all possible non-overlapping patches of a specific size. This experiment aimed to understand whether the improvement in classification performance was due to the patch size,

the cardinality of the set, or both. Since it became evident from previous results that using smaller patch sizes leads to better localization, we focused on patch sizes of 32x32 and 64x64. Below are the results obtained for both classification and localization tasks.

Image Shape	Spatial Resolution	Patch Size	K Patches	Accuracy	Precision	Recall	F1-Score
512x512	50 cm/pixel	32x32	3	63.61%	47.70%	98.27%	64.23%
512x512	50 cm/pixel	32x32	57	83.22%	68.41%	92.03%	78.48%
512x512	50 cm/pixel	32x32	256	81.69%	67.93%	85.10%	75.55%
512x512	50 cm/pixel	64x64	3	84.18%	69.98%	91.80%	79.42%
512x512	50 cm/pixel	64x64	15	86.22%	75.48%	86.72%	80.71%
512x512	50 cm/pixel	64x64	64	82.57%	68.01%	89.84%	77.41%
1024x1024	20 cm/pixel	32x32	3	51.40%	40.51%	98.61%	57.43%
1024x1024	20 cm/pixel	32x32	346	81.77%	66.61%	90.53%	76.75%
1024x1024	20 cm/pixel	32x32	1024	81.54%	74.97%	66.74%	70.62%
1024x1024	20 cm/pixel	64x64	3	64.45%	48.30%	98.38%	64.79%
1024x1024	20 cm/pixel	64x64	86	84.15%	70.17%	90.99%	79.24%
1024x1024	20 cm/pixel	64x64	256	81.65%	73.10%	70.90%	71.98%

Table 4.11: Classification results obtained with ResNet50 using patches with spatial resolutions of 50 and 20 cm/pixel and patch size 32x32 or 64x64.

As evident from the results of the above experiment, the cardinality of the bag of patches impacts classification performance. Particularly, a too small bag fails to capture all the necessary information for correct image classification, while a too large bag includes irrelevant information, making it harder for the model to identify the true features relevant for classification due to the high number of examples to decide among. Below, we proceed to evaluate localization using the parameters that allowed us to achieve the best classification results:

Image Shape	Spatial Resolution	Patch Size	K Patches	IoU	Dice	Precision	Recall	F1-Score
512x512	50 cm/pixel	32x32	57	20.84%	32.12%	30.64%	48.97%	37.70%
512x512	50 cm/pixel	64x64	15	19.44%	30.15%	26.69%	50.62%	34.96%
1024x1024	20 cm/pixel	32x32	346	26.45%	39.53%	46.56%	43.64%	45.06%
1024x1024	20 cm/pixel	64x64	86	24.79%	37.55%	37.63%	51.24%	43.39%

Table 4.12: Localization results obtained with ResNet50 using patches with spatial resolutions of 50 and 20 cm/pixel and patch size 32x32 or 64x64.

As observed from the localization results, it also benefits from having a bag cardinality that is neither too small to encompass the right amount of information nor too high to include superfluous information that complicates the training process. In addition, it remains a

characteristic that the best localization results are achieved with the highest resolution of 20 cm/pixel, thanks to the increased level of detail obtainable at such resolution for identifying the contours of waste.

Figure 4.3: The segmentation masks obtained using the MIL-based approach, respectively, with the former being the original reference mask.

5 | Conclusions and future developments

From this research, it is evident that determining appropriate pre-processing for satellite images is crucial. Specifically, utilizing the maximum spatial resolution has shown to yield better performance due to higher image detail quality. However, it has also been highlighted that not all architectures are capable of achieving optimal performance under these conditions. Therefore, employing high resolutions requires architectures with greater architectural and computational complexity, whereas opting for greater efficiency with medium-sized models necessitates considering their limitations and supported spatial resolutions.

Another fundamental aspect is Data Augmentation, which helps mitigate the challenge of data scarcity. Particularly, the Patch Obscuring transformation enhances heatmaps by compelling models to focus not only on the most discriminative regions of an image.

Regarding the heatmap-based approach, it is clear that extracting GRAD-CAM++ solely from the last feature map of a CNN allows for the localization of waste areas but not for determining their boundaries. This issue has been alleviated by extracting feature maps at different levels: from early layers, better object boundary identification is possible, while with the last feature maps, the area of interest is localized.

Lastly, the MIL-based approach has highlighted the possibility of training models with which inference can be made using patches of reduced dimensions compared to the original large images. However, this approach presents several challenges, foremost among them being the reduction of set cardinality and the identification of relevant patches.

Future developments in this context call for a targeted and innovative approach to overcome current challenges and enhance model performance. The need to accurately pinpoint the factors limiting the results of the ResNet50 at a resolution of 50 cm/pixel is crucial. If resolving these issues proves impossible, exploring a new backbone architecture capable of achieving excellent results in both classification and localization at a 20 cm/pixel resolution is necessary, while balancing computational complexity with efficiency to attain

optimal performance.

Another intriguing aspect that warrants further exploration is the impact of the global context on classification and localization performance. With images at 20 cm/pixel resulting in larger dimensions such as 1088x1088, computational challenges arise due to memory constraints and training time. Thus, investigating the feasibility of using smaller images with a reduced background/context is imperative. However, this poses significant challenges, as the exact locations of waste within images are unknown, making it difficult to ensure that waste remains within cropped images and potentially leading to the generation of false labels.

As for the heatmap-based approach, despite its successes in both tasks, challenges remain, particularly regarding segmentation. Improved identification of precise waste boundaries is necessary, hindered by low-resolution feature maps due to stride > 2 in CNNs. Additionally, current methods tend to highlight only the most discriminative regions in images for predicting the class, rather than capturing all waste instances or the entirety of the same waste object. To address these issues, architectures such as DeepLab could be employed, leveraging Atrous (Dilated) Convolutions and Atrous Spatial Pyramid Pooling to control receptive field and feature map resolutions without inflating the total number of parameters. Furthermore, in numerous literature papers, attempts are made to extract all relevant elements in an image, not just the most discriminative regions, utilizing mining/iterative approaches.

On the other hand, the MIL-based approach offers improved waste boundary detection by utilizing very small patches and identifying all waste within an area. However, the challenge lies in enhancing its precision compared to the previous approach. This challenge arises because each patch is processed independently, as is also the case for the importance weight of each patch. Therefore, improving such architecture involves incorporating relevant information, such as the position of each patch relative to the original image and other patches, to determine adjacent patches where waste may have been split or assess the importance of a patch considering both nearby and distant patches for local and global context, respectively. These issues have been addressed in literature, particularly in Transformers, by introducing position embeddings for the position of each patch and Multi-Head attentions to determine patch information and importance using information from other patches as well.

Bibliography

[1] W. Ahmad, R. D. Alharthy, M. Zubair, M. Ahmed, A. Hameed, and S. Rafique. Toxic and heavy metals contamination assessment in soil and water to evaluate human health risk. *Scientific reports*, 11(1):17006, 2021.

[2] J. Ahn, S. Cho, and S. Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations, 2019.

[3] J. Ba and B. Frey. Adaptive dropout for training deep neural networks. *Advances in neural information processing systems*, 26, 2013.

[4] R. C. Bunescu and R. J. Mooney. Multiple instance learning for sparse positive bags. In *Proceedings of the 24th international conference on Machine learning*, pages 105–112, 2007.

[5] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection, 2017.

[6] M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon. Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, 77:329–353, May 2018. ISSN 0031-3203. doi: 10.1016/j.patcog.2017.10.009. URL `http://dx.doi.org/10.1016/j.patcog.2017.10.009`.

[7] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Mar. 2018. doi: 10.1109/wacv.2018.00097. URL `http://dx.doi.org/10.1109/WACV.2018.00097`.

[8] P. Cunningham, M. Cord, and S. J. Delany. Supervised learning. In *Machine learning techniques for multimedia: case studies on organization and retrieval*, pages 21–49. Springer, 2008.

[9] A. K. Dubey and V. Jain. Comparative study of convolution neural network's relu and leaky-relu activation functions. In *Applications of Computing, Automation and*

Wireless Systems in Electrical Engineering: Proceedings of MARC 2018, pages 873–880. Springer, 2019.

[10] A. V. Etten, D. Lindenbaum, and T. M. Bacastow. Spacenet: A remote sensing dataset and challenge series, 2019.

[11] European Commission CORDIS. PERIVALLON Protecting the EuRopean terrItory from organised enVironmentAl crime through inteLLigent threat detectiON tools. `https://cordis.europa.eu/project/id/101073952`, 2022-2025.

[12] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59:167–181, 2004.

[13] N. Ferronato and V. Torretta. Waste mismanagement in developing countries: A review of global issues. *International journal of environmental research and public health*, 16(6):1060, 2019.

[14] P. Fraternali, L. Morandini, and S. L. H. González. Solid waste detection in remote sensing images: A survey, 2024.

[15] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36 (4):193–202, 1980.

[16] Z. Ghahramani. Unsupervised learning. In *Summer school on machine learning*, pages 72–112. Springer, 2003.

[17] W. Guo, W. Yang, H. Zhang, and G. Hua. Geospatial object detection in high resolution satellite images based on multi-scale convolutional neural network. *Remote Sensing*, 10(1), 2018. ISSN 2072-4292. doi: 10.3390/rs10010131. URL `https://www.mdpi.com/2072-4292/10/1/131`.

[18] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew. Deep learning for visual understanding: A review. *Neurocomputing*, 187:27–48, 2016. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2015.09.116. URL `https://www.sciencedirect.com/science/article/pii/S0925231215017634`. Recent Developments on Deep Big Vision.

[19] Y. Guo, Y. Liu, T. Georgiou, and M. S. Lew. A review of semantic segmentation using deep neural networks. *International journal of multimedia information retrieval*, 7: 87–93, 2018.

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[21] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017. doi: 10.1109/ICCV.2017.322.

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn, 2018.

[23] P. Helber, B. Bischke, A. Dengel, and D. Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification, 2019.

[24] M. Ilse, J. M. Tomczak, and M. Welling. Attention-based deep multiple instance learning, 2018.

[25] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[26] A. Kolesnikov and C. H. Lampert. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 695–711. Springer, 2016.

[27] S. Kwak, S. Hong, and B. Han. Weakly supervised semantic segmentation using superpixel pooling network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[28] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. doi: 10.1162/neco.1989.1.4.541.

[29] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[30] K. Li, G. Wan, G. Cheng, L. Meng, and J. Han. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS Journal of Photogrammetry and Remote Sensing*, 159:296–307, Jan. 2020. ISSN 0924-2716. doi: 10.1016/j.isprsjprs.2019.11.023. URL `http://dx.doi.org/10.1016/j.isprsjprs.2019.11.023`.

[31] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Doll'a r, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL `http://arxiv.org/abs/1405.0312`.

[32] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection, 2018.

[33] K. Liu, Y. Shen, N. Wu, J. Chłędowski, C. Fernandez-Granda, and K. J. Geras. Weakly-supervised high-resolution segmentation of mammography images for breast cancer diagnosis, 2021.

[34] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128:261–318, 2020.

[35] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. *SSD: Single Shot MultiBox Detector*, page 21–37. Springer International Publishing, 2016. ISBN 9783319464480. doi: 10.1007/978-3-319-46448-0_2. URL `http://dx.doi.org/10.1007/978-3-319-46448-0_2`.

[36] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[37] S. Manfredi, D. Tonini, T. H. Christensen, and H. Scharff. Landfilling of waste: accounting of greenhouse gases and global warming contributions. *Waste Management & Research*, 27(8):825–836, 2009.

[38] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3523–3542, 2021.

[39] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[40] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Is object localization for free? - weakly-supervised learning with convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 685–694, 2015. doi: 10.1109/CVPR.2015.7298668.

[41] S.-Y. Pan, C.-Y. Lu, S.-P. Lee, and W.-H. Peng. Weakly-supervised image semantic segmentation using graph convolutional networks, 2021.

[42] PERIVALLON. Introduction to PERIVALLON Project. `https://perivallon-he.eu/introduction-to-perivallon-project-2/`, 2022-2025.

[43] H. Ramsauer, B. Schäfl, J. Lehner, P. Seidl, M. Widrich, T. Adler, L. Gruber, M. Holzleitner, M. Pavlović, G. K. Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

[44] W. Rawat and Z. Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449, 2017.

[45] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection, 2016.

[46] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016.

[47] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[48] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[49] S. Sharma, S. Sharma, and A. Athaiya. Activation functions in neural networks. *Towards Data Sci*, 6(12):310–316, 2017.

[50] K. K. Singh and Y. J. Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization, 2017.

[51] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions, 2014.

[52] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning, 2016.

[53] R. N. Torres and P. Fraternali. Aerialwaste dataset for landfill discovery in aerial and satellite images. *Scientific Data*, 10(1):63, Jan 2023. ISSN 2052-4463. doi: 10.1038/s41597-023-01976-9. URL `https://doi.org/10.1038/s41597-023-01976-9`.

[54] A. Voulodimos, N. Doulamis, A. Doulamis, E. Protopapadakis, et al. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

[55] Q. Wang, X. Liu, Y. Cui, Y. Tang, W. Chen, S. Li, H. Yu, Y. Pan, and C. Wang. The e3 ubiquitin ligase amfr and insig1 bridge the activation of tbk1 kinase by modifying the adaptor sting. *Immunity*, 41(6):919–933, 2014.

[56] S. Yang, Y. Kim, Y. Kim, and C. Kim. Combinational class activation maps for weakly supervised object localization, 2019.

[57] Y. Yang and S. Newsam. Uc merced land use dataset. URL `http://vision.ucmerced.edu/datasets/landuse.html`.

[58] B. Zhang, J. Xiao, Y. Wei, M. Sun, and K. Huang. Reliability does matter: An end-to-end weakly supervised semantic segmentation approach, 2019.

[59] D. Zhang, J. Han, G. Cheng, and M.-H. Yang. Weakly supervised object localization and detection: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5866–5885, 2021.

[60] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

[61] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu. Scale-transferrable object detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 528–537, 2018. doi: 10.1109/CVPR.2018.00062.

[62] Z.-H. Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53, 08 2017. ISSN 2095-5138. doi: 10.1093/nsr/nwx106. URL `https://doi.org/10.1093/nsr/nwx106`.

[63] X. J. Zhu. Semi-supervised learning literature survey. 2005.

List of Figures

List of Tables

List of Algorithms

6 | Ringraziamenti

Desidero esprimere la mia profonda gratitudine al Professor Giacomo Boracchi per la sua guida preziosa, la sua pazienza infinita e il suo sostegno costante durante il percorso di ricerca e scrittura di questa tesi.

Un ringraziamento speciale va al Professor Piero Fraternali per avermi offerto l'opportunità di partecipare a questa incredibile esperienza di ricerca.

Desidero anche ringraziare il dottorando Andrea per il suo sostegno instancabile. La sua disponibilità e la sua dedizione hanno reso possibile affrontare le sfide e superare gli ostacoli lungo il cammino.

Un sentito ringraziamento va ai miei colleghi tesisti Elena e Federico, con cui ho avuto il privilegio di collaborare.

Infine, vorrei ringraziare i miei amici e i miei genitori per il loro amore, il loro incoraggiamento e il loro costante sostegno. Senza di voi, questo traguardo non sarebbe stato possibile.